

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

ÉCOLE NATIONALE POLYTECHNIQUE



المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique

DÉPARTEMENT D'ÉLECTRONIQUE

Mémoire de projet de fin d'études pour l'obtention du diplôme
d'ingénieur d'état en Électronique

Thème :

**Décodage des codes LDPC par algorithme Gradient
Descent Bit Flipping (GDBF) à mémoire d'état**

Nabila RAHAL

Sous la direction de M. Mohamed Oussaïd TAGHI

Présenté et soutenu publiquement le 22/06/2026

Composition du Jury :

Président	Pr. Cherif LARBES	ENP
Promoteur	M. Mohamed Oussaïd TAGHI	ENP
Examineur	Dr. Salah BOUHOUN	ENP

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

ÉCOLE NATIONALE POLYTECHNIQUE



المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique

DÉPARTEMENT D'ÉLECTRONIQUE

Mémoire de projet de fin d'études pour l'obtention du diplôme
d'ingénieur d'état en Électronique

Thème :

**Décodage des codes LDPC par algorithme Gradient
Descent Bit Flipping (GDBF) à mémoire d'état**

Nabila RAHAL

Sous la direction de M. Mohamed Oussaïd TAGHI

Présenté et soutenu publiquement le 22/06/2026

Composition du Jury :

Président	Pr. Cherif LARBES	ENP
Promoteur	M. Mohamed Oussaïd TAGHI	ENP
Examineur	Dr. Salah BOUHOUN	ENP

Dédicaces

Avec l'expression de ma plus profonde reconnaissance, je dédie ce modeste travail à ceux pour qui les mots ne suffiront jamais à exprimer tout mon amour et ma gratitude.

À ma très chère mère,

La femme qui a souffert sans me laisser souffrir, qui n'a jamais dit non à mes exigences et qui n'a cessé de prier pour moi. Ma source d'espoir, qui n'a épargné aucun effort pour me rendre heureuse.

À mon très cher père,

L'homme qui m'a offert la vie et à qui je dois ma réussite. Pour ton amour, tes sacrifices, tes encouragements et ton soutien infaillible afin que rien n'entrave le déroulement de mes études. Tu as tout mon respect.

À mon cher amour, mon mari,

Toi qui m'as toujours aidée, soutenue et motivée par tes précieux conseils. Merci d'avoir été mon pilier et de m'avoir toujours rassurée en me disant : « Je suis avec toi, ne baisse pas les bras ».

À ma chère sœur LINA et ma cousine ROFEIDA,

Pour votre présence chaleureuse, votre affection et vos encouragements tout au long de mon parcours. Que Dieu vous protège et vous offre la chance et le bonheur.

À tous mes amis,

Qui ont partagé avec moi les moments de joie, de rires, mais aussi de stress et de travail acharné durant toutes ces années d'études.

Remerciements

En premier lieu, je remercie Dieu, le Tout-Puissant, de m'avoir donné le courage, la volonté et la patience pour entamer et réaliser ce travail.

*Je tiens à exprimer ma profonde gratitude à mon promoteur, **Monsieur Mohamed TAGHI**, pour avoir accepté de diriger ce projet. Je le remercie pour ses conseils précieux, ses orientations scientifiques et la confiance qu'il m'a accordée tout au long de la réalisation de ce mémoire.*

*Je tiens également à remercier chaleureusement les membres qui composent mon jury pour l'honneur qu'ils me font en évaluant mon travail : **Pr. LARBES Cherifi**, d'avoir accepté de présider ce jury, et **Dr. BOUHOUN Salah**, pour l'intérêt qu'il a porté à ce travail en acceptant de l'examiner.*

Mes remerciements s'adressent également à tous mes professeurs du département d'Électronique de l'École Nationale Polytechnique pour leur générosité, la qualité de leur enseignement et la grande patience dont ils ont fait preuve malgré leurs lourdes charges académiques.

Enfin, un grand merci à toutes les personnes qui ont contribué, de près ou de loin, par leur aide pratique ou leur soutien moral, à l'aboutissement de ce projet de fin d'études.

ملخص

تعتبر رموز بضعت أساسية في أنظمة الاتصالات الحديثة. بعد وصف مبسط للمشفّر، نركز على تقنية فك التشفير بقلب البت (بذ). رغم تعقيدها المادي المنخفض جداً، تعاني من محدودية الأداء بسبب المجموعات الفخية. لحل ذلك، نقترح طريقة مبتكرة تعتمد على التعلم الآلي لتحسين خوارزمية جضد باستخدام آلة حالات محدودة (ذص). أظهرت المحاكاة المسرعة عبر جعو تفوق طريقتنا على التقنيات التقليدية، حيث تتخلص من المجموعات الفخية وتقدم أداءً ممتازاً مع الحفاظ على تعقيد مادي منخفض.

الكلمات المفتاحية : بضعت، المشفر، قلب البت، التعلم الآلي، جضد، ذص، تسريع جعو.

Abstract

LDPC codes are vital in modern communication systems. Following a brief encoder description, this work focuses on Bit Flipping (BF) decoding. While BF ensures very low hardware complexity, it suffers from error floors caused by trapping sets. To resolve this, we propose an innovative Machine Learning approach enhancing the GDBF algorithm with a Finite State Machine (FSM). GPU-accelerated simulations demonstrate that our method outperforms conventional techniques, successfully escaping trapping sets and achieving near-optimal performance with minimal hardware complexity.

Keywords : LDPC, Encoder, Bit Flipping, Machine Learning, GDBF, FSM, GPU.

Résumé

Les codes LDPC sont essentiels dans les systèmes de communication modernes. Après une description succincte du codeur, ce travail analyse le décodage *Bit Flipping* (BF). Bien que le BF offre une très faible complexité matérielle, il est limité par un plancher d'erreur dû aux ensembles pièges. Pour y remédier, nous proposons une méthode innovante d'apprentissage automatique améliorant l'algorithme GDBF via une Machine à États Finis (FSM). Les simulations accélérées par GPU prouvent que notre approche surpasse les techniques conventionnelles en éliminant les ensembles pièges, offrant d'excellentes performances tout en maintenant une faible complexité matérielle.

Mots-clés : LDPC, Codeur, Bit Flipping, Apprentissage automatique, GDBF, FSM, GPU.

Table des matières

Liste des tableaux

Table des figures

Liste des Abréviations

Liste des Symboles

Introduction Générale	14
1 Généralités sur le codage canal et les Codes LDPC	16
1.1 Introduction	17
1.2 La chaîne de communication numérique	17
1.2.1 Le codage de source	18
1.2.2 Le codage de canal	18
1.2.3 La modulation numérique	19
1.2.4 Le canal de communication	19
1.2.5 La capacité d'un canal	19
1.2.6 Le théorème fondamental du codage de canal	20
1.3 Fondements théoriques du codage correcteur d'erreurs	21
1.3.1 Principes et caractéristiques des codes en bloc	21
1.3.2 Les codes convolutifs	23
1.3.3 Les Turbo-codes	23
1.3.4 Les codes polaires	24
1.3.5 Comparaison des performances	25
1.4 Conclusion du chapitre	26
2 Structure des codes LDPC	28
2.1 Introduction	29
2.2 Fondements des codes Low-Density Parity-Check (LDPC)	29
2.2.1 Définition algébrique et historique	30

2.2.2	La matrice de contrôle de parité H et classification	31
2.2.3	Représentation topologique : Le graphe de Tanner	32
2.2.4	Les codes LDPC Quasi-Cycliques (QC-LDPC)	33
2.3	Algorithmes de décodage itératifs classiques	35
2.3.1	Le décodage à décision stricte (Hard-Decision)	35
2.3.2	Le décodage à décision pondérée (Soft-Decision)	35
2.4	L'algorithme Sum-Product (SPA) / Belief Propagation	36
2.5	L'Algorithme Bit Flipping (BF)	38
2.6	Comparaison des décodeurs et justification du choix	39
2.7	Le Gradient Descent Bit Flipping (GDBF)	40
2.8	Problématique et Transition vers le Machine Learning	43
2.9	Conclusion du chapitre	44
3	Décodage GDBF à Machine à États Finis : Architecture et Optimisation des Paramètres	45
3.1	Introduction	46
3.2	État de l'art du décodage Bit-Flipping par Apprentissage Automatique . .	46
3.3	Évolution historique et genèse de l'algorithme GDBF-FSM	47
3.4	Architecture du décodeur GDBF-FSM	48
3.4.1	Définition de la Machine à États Finis	50
3.4.2	Exemple de résolution d'un ensemble piège (<i>Trapping Set</i>)	51
3.4.3	Framework d'Apprentissage et d'Optimisation des Paramètres . . .	53
3.5	Étude des performances de l'algorithme	55
3.5.1	Matrice de Tanner	55
3.5.2	Matrice i-RISC	57
3.6	Conclusion du chapitre	58
4	Simulations et évaluation des performances	60
4.1	Introduction	61
4.2	Présentation de la chaîne de simulation	61
4.3	Résultats du décodage classique sur la matrice DVB-S2	62
4.4	Évaluation de l'algorithme gAD-GDBFwM-wSM sur la matrice de Tanner .	63
4.5	Évaluation de l'algorithme gAD-GDBFwM-wSM sur la matrice de i-RISC	65
4.6	Synthèse comparative des résultats	67
4.7	Conclusion du chapitre	68
	Conclusion Générale	69
	Bibliographie	71

Liste des tableaux

2.1	Comparaison entre le décodage à décision pondérée (SPA) et stricte (BF) .	40
4.1	Synthèse comparative des performances (FER) des décodeurs étudiés sur les trois matrices LDPC testées.	67

Table des figures

1.1	Schéma simplifié d'une chaîne de communications numériques.	18
1.2	Architecture générale d'un système de codage et décodage en bloc linéaire.	22
1.3	Codeur convolutif de fonction de transfert $[1, 1 + D + D^3]$	23
1.4	Représentation du code convolutif sous forme de diagramme en treillis.	23
1.5	Architecture d'un système de Turbo-codage basé sur la concaténation parallèle.	24
1.6	Processus de Turbo-décodage itératif avec échange d'informations extrinsèques.	25
1.7	Comparaison des performances entre Turbo-codes, codes LDPC et codes polaires	26
2.1	Représentation topologique d'un code LDPC via un graphe bipartite de Tanner mettant en évidence le routage entre nœuds de variable (VN) et nœuds de contrôle (CN).	33
2.2	Structure macro-matricielle d'un code QC-LDPC de dimensions 448160 pour un facteur d'expansion $Z = 32$	34
2.3	Organigramme de l'algorithme de décodage Sum-Product (BP) [6].	37
2.4	Procédure de l'algorithme de décodage Bit Flipping [6].	38
2.5	Évaluation comparative des performances du GDBF face aux algorithmes heuristiques classiques (WBF, MWBF) sur canal AWGN (Tiré de Wadayama et al., 2010 [11]).	41
2.6	Impact du processus d'échappement sur la dynamique de convergence et la mitigation des minima locaux dans l'algorithme GDBF (Tiré de Wadayama et al., 2010 [11]).	42
3.1	Schéma synoptique de l'évolution des algorithmes de décodage, du GDBF classique à l'approche GDBF-FSM.	47
3.2	Algorithme de décodage gAD-GDBFwM FSM [12].	50
3.3	Diagramme de transition des états de la FSM locale (Adapté de Milojković et al., 2025 [12]).	51

3.4	Évolution des paramètres d'un ensemble piège sous l'action de la FSM locale (Adapté de Milojković et al., 2025 [12]).	52
3.5	Organigramme du processus d'apprentissage hors-ligne pour la collecte des erreurs et l'optimisation des paramètres du décodeur [12].	54
3.6	Performances en fonction de L_{max} sur canal BSC pour le code de Tanner (155, 64), avec $\gamma = 3$ et $\rho = 5$ [12].	56
3.7	Performances comparées des différents décodeurs sur canal BSC pour le code de Tanner (155, 64), avec $\gamma = 3$ et $\rho = 5$ [12].	56
3.8	Performances en fonction de L_{max} sur canal BSC pour le code i-RISC (1296, 648), avec $\gamma = 4$ et $\rho = 8$ [12].	57
3.9	Performances en fonction de L_{max} sur canal AWGN pour le code i-RISC (1296, 648), avec $\gamma = 4$ et $\rho = 8$ [12].	58
4.1	Modèle de la chaîne de simulation de transmission LDPC et évaluation des performances.	62
4.2	Performances des décodeurs BF et GDBF sur la matrice DVB-S2 ($R = 0.5$) sur canal AWGN.	63
4.3	Évolution du Taux d'Erreur Trame (FER) en fonction de L_{max} pour le code de Tanner sur canal BSC ($\alpha = 0.01$)	64
4.4	Évolution du FER en fonction de la probabilité de croisement α pour le code i-RISC (1296, 648) sur canal BSC.	65
4.5	Évolution du FER en fonction du SNR pour le code i-RISC (1296, 648) sur canal AWGN.	66

Liste des Abréviations

- **AGDBF** : Adaptive Gradient Descent Bit Flipping
- **ASIC** : Application-Specific Integrated Circuit
- **AWGN** : Additive White Gaussian Noise
- **BCH** : Bose-Chaudhuri-Hocquenghem
- **BER** : Bit Error Rate
- **BF** : Bit Flipping
- **BP** : Belief Propagation
- **BPSK** : Binary Phase-Shift Keying
- **BSC** : Binary Symmetric Channel
- **CN** : Check Nodes
- **DGDBF** : Diversity Gradient Descent Bit Flipping
- **DVB-S2** : Digital Video Broadcasting - Satellite - Second Generation
- **FER** : Frame Error Rate
- **FPGA** : Field-Programmable Gate Array
- **FSM** : Finite State Machine
- **GA** : Genetic Algorithm
- **gAD-GDBFwM-wSM** : Generalized Adaptive Diversity Gradient Descent Bit-Flipping with Momentum and State Machine
- **GDBF** : Gradient Descent Bit Flipping
- **LDPC** : Low-Density Parity-Check
- **LLR** : Log-Likelihood Ratio
- **MDP** : Markov Decision Process
- **MWBF** : Modified Weighted Bit Flipping
- **NR** : New Radio
- **PGDBF** : Probabilistic Gradient Descent Bit Flipping
- **QC-LDPC** : Quasi-Cyclic Low-Density Parity-Check
- **SNR** : Signal-to-Noise Ratio
- **SPA** : Sum-Product Algorithm
- **TRGDBF** : Threshold-Refinement Gradient Descent Bit Flipping
- **VN** : Variable Nodes

— **WBF** : Weighted Bit Flipping

Liste des Symboles

n	Longueur du mot de code
k	Nombre de bits d'information utiles
R	Rendement du code de canal
$\mathbf{H}_{mn}$	Matrice de contrôle de parité de dimensions mn
$h_{i,j}$	Élément binaire de la matrice de contrôle de parité
v_i	Nœud de variable (VN) associé à la i -ème colonne
c_j	Équation de parité (Nœud de contrôle - CN) liée à la j -ème ligne
$P(v_i)$	Ensemble des indices des nœuds de contrôle connectés à v_i
$Q(c_j)$	Ensemble des indices des nœuds de variable connectés à c_j
γ	Degré du nœud de variable v_i (pour un code LDPC régulier)
ρ	Degré du nœud de contrôle c_j (pour un code LDPC régulier)
$\mathbf{x}$	Vecteur du mot de code transmis
$\mathbf{y}$	Vecteur du signal intercepté (mot de code reçu)
$\hat{\mathbf{x}}$	Vecteur du mot de code estimé par le décodeur
$L_{max,t}$	Nombre maximum d'itérations autorisées pour le t -ème décodeur
L_{max}	Nombre maximum d'itérations pour la chaîne de décodeurs complète
$r_{flag}^{(t)}$	Indicateur de redémarrage (<i>restart flag</i>) du t -ème décodeur
$E_i^{(\ell)}$	Fonction d'énergie pour le i -ème nœud de variable à la ℓ -ème itération
w_1, w_2	Poids d'optimisation apprenables (<i>learnable weights</i>)
m_{μ_i}	Valeur de l'impulsion (<i>momentum</i>) pour la variable i
μ_i	État de l'impulsion (<i>momentum state</i>) de la i -ème variable
$\mathbf{m}$	Vecteur d'impulsion (<i>momentum vector</i>)
L'	Nombre de valeurs discrètes dans le vecteur de momentum
I	Valeur maximale dans le vecteur de momentum
$\mathcal{F}^{(\ell)}$	Ensemble des variables mises en évidence (suspectes) à la ℓ -ème itération
δ	Paramètre de marge de tolérance ou de seuil
$r_i^{(\ell)}$	Potentiel local de la variable v_i à la ℓ -ème itération
θ	Valeur de seuil d'activation pour la règle θ
$\mathbf{r}_{in}^{(t)}$	Vecteur des potentiels d'entrée pour le t -ème décodeur

$\mu_{in}^{(t)}$	Vecteur des états de momentum initiaux pour le t -ème décodeur
$\mathbf{p}_{in}^{(t)}$	Vecteur d'états d'initialisation pour la machine à états finis
S	Modèle de la Machine à États Finis (FSM)
S	État spécifique dans la machine à états finis
s	Valeur numérique de l'état dans la machine à états finis
S_+	Ensemble des états positifs dans la machine à états finis
η	Nombre total d'états dans l'ensemble S_+
S_-	Ensemble des états négatifs dans la machine à états finis
ζ	Nombre total d'états dans l'ensemble S_-
S_0	État neutre de la FSM (zone d'hésitation)
S_{start}	État initial de démarrage de la FSM
$\mathcal{S}_t$	Fonction de conversion ou de transition des machines à états finis

Introduction Générale

Les réseaux sans fil modernes (5G, communications par satellite) exigent des transmissions à la fois très rapides et fiables. Pour protéger les données contre le bruit et les interférences du canal, on utilise le codage de canal. Le principe est d'ajouter des bits de redondance au message avant l'émission, ce qui permet au récepteur de détecter et de corriger les erreurs de manière autonome.

Parmi les solutions existantes, les codes LDPC (*Low-Density Parity-Check*) sont devenus un standard incontournable grâce à leurs excellentes capacités de correction. Cependant, leur implémentation matérielle pose un vrai problème technique. L'algorithme de décodage idéal, le *Belief Propagation* (BP), demande beaucoup trop de calculs et de ressources mémoire, ce qui complique son utilisation en temps réel.

L'alternative pratique pour l'industrie consiste à utiliser des algorithmes à décision stricte, comme le *Bit Flipping* (BF). Ils sont simples et très légers à intégrer sur des circuits électroniques. Le problème, c'est que ce type de décodeur se bloque souvent dans des configurations mathématiques appelées ensembles pièges (*Trapping Sets*). L'algorithme tourne en boucle sur de faux minima locaux et crée un plancher d'erreur (*Error Floor*) qu'il n'arrive pas à corriger.

L'objectif de ce projet de fin d'études est de résoudre ce problème de blocage tout en gardant la simplicité matérielle du Bit Flipping. Au lieu d'alourdir le circuit logique, nous avons choisi une approche logicielle basée sur le *Machine Learning*. L'idée est d'améliorer l'algorithme *Gradient Descent Bit Flipping* (GDBF) en lui intégrant une Machine à États Finis (FSM). Les paramètres de cette FSM sont optimisés hors-ligne. Ainsi, le décodeur devient capable de s'extraire des minima locaux de façon déterministe. Pour valider notre approche, tout le système a été codé et simulé sur MATLAB, avec une accélération sur carte graphique (GPU) pour traiter rapidement les simulations de Monte-Carlo.

Ce mémoire est organisé en quatre chapitres :

Le Chapitre 1 fait un rappel sur les concepts de base de la théorie de l'information

(canal de transmission, limite de Shannon) et présente les principales familles de codes correcteurs d'erreurs.

Le Chapitre 2 est consacré à l'étude des codes LDPC. Nous détaillons leur construction (matrice de parité, graphe de Tanner) et nous analysons les limites de l'algorithme GDBF classique face au phénomène des ensembles pièges.

Le Chapitre 3 présente notre solution de décodage optimisé. Nous expliquons le fonctionnement de l'architecture GDBF couplée à la FSM, ainsi que la méthode d'apprentissage hors-ligne par algorithme génétique.

Le Chapitre 4 regroupe les résultats de nos simulations. Nous comparons les performances (taux d'erreur BER/FER) de notre modèle par rapport aux algorithmes classiques sur plusieurs matrices (Tanner, i-RISC), tout en discutant de l'impact sur la complexité matérielle.

Chapitre 1

Généralités sur le codage canal et les Codes LDPC

1.1 Introduction

Ce premier chapitre définit le cadre mathématique d’une transmission numérique. Il modélise l’architecture standard d’une chaîne de communication, avec une focale stricte sur le bloc de codage de canal. Les métriques fondamentales de la théorie de l’information (rendement, capacité, rapport signal sur bruit) sont ici exposées. Elles serviront de référentiel analytique pour évaluer les performances des codes à contrôle de parité à faible densité (LDPC) et de leurs décodeurs itératifs.

Une liaison de télécommunication a pour fonction de transférer un flux binaire depuis un émetteur vers un récepteur à travers un canal de transmission. Ce milieu physique altère systématiquement les signaux par l’ajout de bruit stochastique (généralement modélisé par un bruit blanc gaussien additif, AWGN) ou par des phénomènes d’atténuation. L’objectif technique n’est pas de supprimer ces perturbations physiques, mais de concevoir un système de traitement capable d’estimer la séquence binaire initiale avec une probabilité d’erreur minimale à la réception.

Les premiers modèles d’évaluation de la bande passante établis par Nyquist [1] et Hartley [2] ont été mathématisés de façon rigoureuse en 1948 par Claude Shannon [3]. Son théorème du codage de canal définit une limite théorique stricte, appelée capacité du canal (C). Shannon démontre que si le débit binaire de transmission R est inférieur à cette capacité ($R < C$), il existe un code correcteur permettant d’atteindre un taux d’erreur binaire (BER) proche de zéro. La mise en pratique de ce postulat repose sur le codage de canal. Le procédé consiste à injecter des bits de parité calculés algébriquement dans le flux de données avant modulation, fournissant ainsi au décodeur la redondance nécessaire pour détecter et corriger les bits erronés.

1.2 La chaîne de communication numérique

Une chaîne de communication numérique modélise les blocs de traitement nécessaires au transfert d’information entre une source et un destinataire. Bien que l’architecture matérielle diffère selon l’application finale (système de stockage de données ou télécommunications spatiales), la topologie du traitement du signal reste standardisée.

Ce traitement s’articule autour de deux blocs aux objectifs opposés. En amont, le codeur de source compresse les données pour minimiser le débit binaire et optimiser la bande passante. À l’inverse, le codeur de canal augmente volontairement la taille du message. L’insertion de cette redondance algébrique constitue l’unique mécanisme matériel et logiciel permettant de fiabiliser la liaison face aux erreurs de transmission.

Le canal de communication ne se limite pas au seul support physique de propagation. Il englobe l’ensemble du trajet matériel séparant la sortie de l’encodeur de l’entrée du déco-

deur. Ce segment intègre de manière critique les étages de modulation et de démodulation. Ces deux sous-systèmes assurent la transposition de la séquence binaire discrète en un signal analogique continu, façonné pour correspondre aux caractéristiques fréquentielles du milieu physique, avant d'opérer la conversion inverse à la réception [4].

Cette architecture canonique, illustrée par la Figure 1.1, s'inscrit rigoureusement dans le modèle théorique de Shannon. Le canal y est défini comme l'interface physique reliant deux nœuds distants. Ce milieu de propagation est intrinsèquement hostile aux signaux : il dégrade le flux de données émis par l'ajout de bruit thermique, la création d'évanouissements de signal (*fading*) et la génération d'interférences [3, 5].

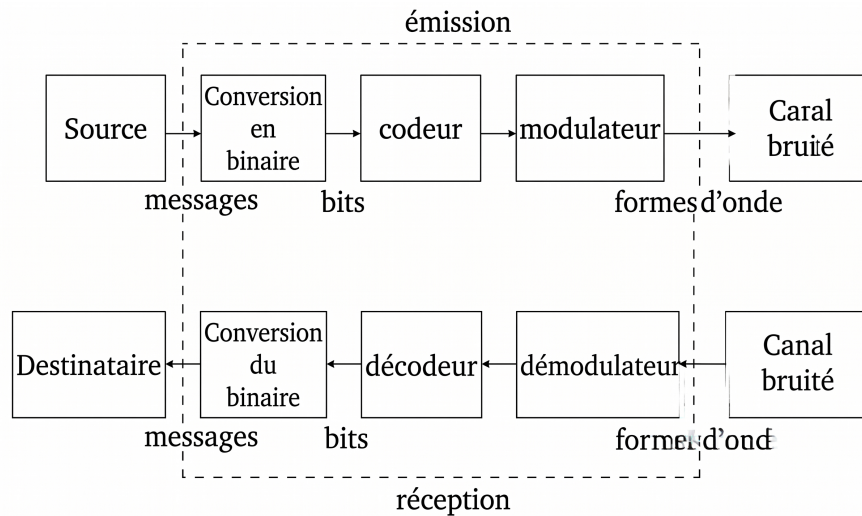


FIGURE 1.1 – Schéma simplifié d'une chaîne de communications numériques.

1.2.1 Le codage de source

L'objectif premier du codage de source est d'optimiser l'exploitation des ressources disponibles (bande passante, temps ou puissance d'émission) en représentant le message initial avec le moins d'éléments binaires possible. Concrètement, cette étape permet de compresser l'information en supprimant simplement sa redondance naturelle [6]. En fonction des contraintes de l'application visée, ce processus peut être réversible (c'est-à-dire sans perte, ce qui garantit une restitution mathématiquement exacte des données) ou bien non-réversible. Cette seconde approche tolère une certaine distorsion du signal afin d'obtenir un gain de compression beaucoup plus important, une méthode particulièrement adaptée à la transmission de flux audio et vidéo.

1.2.2 Le codage de canal

Contrairement au codage de source dont la fonction est la compression, le codage de canal vise à protéger l'information en introduisant volontairement une redondance

mathématique structurée. Cette protection permet d'assurer l'intégrité du message face aux perturbations inhérentes au canal de transmission [4]. Concrètement, pour un bloc source de K bits, le codeur produit un mot de code de longueur N , incluant ainsi $M = N - K$ bits de parité. L'efficacité de cette redondance est quantifiée par le rendement du code, défini par le rapport $R = K/N$. À la réception, le rôle du décodeur est d'exploiter cette structure de parité pour identifier les erreurs et, sous réserve que celles-ci restent dans la capacité de correction du code, de restituer les données originales [6].

1.2.3 La modulation numérique

Les séquences binaires issues du codeur de canal ne peuvent pas se propager telles quelles dans un milieu physique. La modulation numérique réalise une transposition de l'espace discret (binaire) vers l'espace euclidien (continu) [6]. Les éléments binaires sont regroupés par blocs de L bits pour sélectionner un signal analogique de durée T parmi un alphabet de $M = 2^L$ formes d'ondes distinctes (variant en amplitude, phase ou fréquence). Ce signal continu est alors strictement adapté aux caractéristiques fréquentielles du support de transmission.

1.2.4 Le canal de communication

Le canal de communication englobe l'intégralité du milieu physique séparant l'émetteur du récepteur (câble, fibre optique, espace libre). Lors de sa propagation, le signal analogique subit diverses altérations : atténuation de la puissance avec la distance, trajets multiples (fading), interférences et ajout d'un bruit thermique [6]. Le modèle mathématique le plus standard pour évaluer un système de correction d'erreurs est le canal à Bruit Blanc Gaussien Additif (AWGN). Dans ce modèle, le signal reçu correspond à l'addition linéaire du signal émis et d'une variable aléatoire gaussienne indépendante de moyenne nulle [4].

1.2.5 La capacité d'un canal

La transmission d'un signal à travers un canal physique s'accompagne inévitablement de dégradations. Dès lors, une question fondamentale s'est rapidement imposée à la communauté scientifique : existe-t-il une limite physique absolue à la quantité d'information transmissible de manière fiable sur un support donné ?

Si les prémices de cette réflexion remontent aux années 1920 avec les travaux précurseurs de Nyquist [1] et Hartley [2] — reliant déjà largeur de bande et débit télégraphique — c'est en octobre 1948 que la réponse prend toute sa dimension mathématique. Dans son ouvrage fondateur *"A Mathematical Theory of Communication"* [3], Claude E. Shannon a radicalement transformé l'ingénierie en introduisant la définition formelle de "capacité de canal".

Dans le cadre de la théorie de l'information, la capacité C d'un canal discret sans mémoire correspond au maximum de l'information mutuelle $I(X; Y)$ entre la source X et la sortie Y [6], soit :

$$C = \max_{P(X)} I(X, Y) \quad (1.1)$$

Cette valeur, obtenue par maximisation sur l'ensemble des distributions de probabilité d'entrée $P(X)$, quantifie le débit maximal d'information utile capable de traverser le canal en tenant compte du bruit ambiant [6].

Pour les systèmes réels, souvent modélisés comme des canaux à Bruit Blanc Gaussien Additif (AWGN), cette borne est exprimée par la célèbre relation de Shannon-Hartley :

$$C = W \log_2 \left(1 + \frac{S}{N} \right) \quad (1.2)$$

où W désigne la largeur de bande en Hertz, S la puissance moyenne du signal reçu, et N la puissance du bruit [5]. Cette équation souligne la dépendance intrinsèque entre la capacité de transmission, la ressource spectrale (W) et l'efficacité énergétique (S/N).

D'un point de vue physique, cette limite représente une borne infranchissable. Si le débit de la source R dépasse la capacité ($R > C$), le taux d'erreur devient inévitablement inacceptable, quel que soit le schéma de codage utilisé. À l'inverse, si $R < C$, Shannon a démontré qu'il est théoriquement possible d'atteindre une probabilité d'erreur arbitrairement proche de zéro grâce à des codes correcteurs d'erreurs appropriés [3, 4]. C'est précisément l'enjeu des travaux actuels sur les codes LDPC : tendre vers cette limite théorique établie il y a plus de 75 ans.

1.2.6 Le théorème fondamental du codage de canal

Le deuxième théorème de Shannon, ou théorème fondamental du codage de canal, établit la condition stricte régissant toute communication fiable. Il stipule que si le débit d'information R (ou le rendement de codage) est strictement inférieur à la capacité du canal C ($R < C$), il existe mathématiquement des codes correcteurs en bloc permettant d'atteindre une probabilité d'erreur de décodage arbitrairement faible [3, 6]. Ce théorème garantit qu'en définissant une redondance adéquate, la transmission peut résister aux pires perturbations physiques. Ce constat théorique justifie l'existence et la pertinence des familles de codes modernes, telles que les codes LDPC étudiés dans les sections suivantes, qui visent à s'approcher de cette limite fondamentale.

1.3 Fondements théoriques du codage correcteur d'erreurs

Afin de tendre vers les limites théoriques de fiabilité établies par Shannon, le codage correcteur d'erreurs s'est imposé comme une discipline indispensable en télécommunications. Le principe fondamental repose sur l'insertion structurée d'une redondance mathématique au sein du message source. Bien que cette redondance engendre une réduction du rendement effectif du système, elle octroie au récepteur la capacité de détecter et de corriger les altérations induites par le canal de transmission [6]. Avant d'aborder les familles de codes modernes, il est nécessaire de poser les bases de l'architecture algébrique qui sous-tend ces mécanismes : le codage en bloc.

1.3.1 Principes et caractéristiques des codes en bloc

Le codage en bloc repose sur une segmentation rigoureuse du flux de données source en paquets de taille fixe. Concrètement, chaque vecteur d'information $\mathbf{u}$, constitué de K bits, est transformé par l'encodeur en un mot de code $\mathbf{v}$ de longueur N (où $N > K$). Cette extension introduit $M = N - K$ bits de redondance, couramment nommés bits de parité [4].

La performance et l'efficacité spectrale du système sont directement corrélées au **rendement de codage** (R), paramètre fondamental défini par le rapport :

$$R = \frac{K}{N} \quad (1.3)$$

D'un point de vue algébrique, un code en bloc linéaire est caractérisé par deux matrices fondamentales complémentaires [6, 4] :

- **La matrice génératrice $\mathbf{G}$** (de dimensions KN) : Utilisée lors de la phase d'émission pour élaborer le mot de code, l'encodage s'effectue via une multiplication matricielle dans le corps de Galois $GF(2)$, selon l'expression $\mathbf{v} = \mathbf{u} \cdot \mathbf{G}$. Exemple de matrice génératrice :

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \quad (1.4)$$

- **La matrice de contrôle de parité $\mathbf{H}$** (de dimensions MN) : Déployée à la réception, elle permet de valider l'intégrité du message. Par définition, les matrices $\mathbf{G}$ et $\mathbf{H}$ sont orthogonales, respectant ainsi la relation $\mathbf{G} \cdot \mathbf{H}^T = \mathbf{0}$. Par conséquent, tout mot de code valide $\mathbf{v}$ doit impérativement satisfaire l'équation de syndrome

suivante : $\mathbf{v} \cdot \mathbf{H}^T = \mathbf{0}$. Exemple de matrice de contrôle de parité :

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \quad (1.5)$$

Au-delà de ces propriétés matricielles, la robustesse d'un code en bloc est dictée par sa **distance minimale de Hamming** (d_{min}). Cette valeur, correspondant au nombre minimal de positions binaires distinctes entre deux mots de code, conditionne directement la capacité de correction du système. Pour assurer la correction de e erreurs au sein d'un bloc, la condition structurelle $d_{min} \geq 2e + 1$ doit être rigoureusement satisfaite [6].

Calcul du syndrome et détection d'erreurs : Lors de la transmission, le canal bruité peut corrompre le signal émis. Le vecteur intercepté par le récepteur, noté $\mathbf{y}$, se traduit par la somme du mot de code original $\mathbf{v}$ et d'un vecteur d'erreur $\mathbf{e}$ ($\mathbf{y} = \mathbf{v} + \mathbf{e}$). Pour identifier la présence d'altérations, le décodeur calcule un vecteur test appelé **syndrome** ($\mathbf{S}$), défini par la relation :

$$\mathbf{S} = \mathbf{y} \cdot \mathbf{H}^T = (\mathbf{v} + \mathbf{e}) \cdot \mathbf{H}^T = \mathbf{e} \cdot \mathbf{H}^T \quad (1.6)$$

Pour illustrer ce mécanisme, considérons un message source $\mathbf{u} = [0 \ 1 \ 0 \ 1]$. Après codage via la matrice $\mathbf{G}$, le système produit le mot de code $\mathbf{v} = [0 \ 1 \ 0 \ 1 \ 1 \ 0 \ 1]$. En l'absence de perturbation, le calcul du syndrome donne $\mathbf{S} = \mathbf{0}$, confirmant la validité du bloc reçu. À l'inverse, si un bruit canal altère la valeur du dernier bit, le vecteur reçu devient $\mathbf{y} = [0 \ 1 \ 0 \ 1 \ 1 \ 0 \ 0]$. L'opération $\mathbf{y} \cdot \mathbf{H}^T$ génère alors un syndrome non nul (par exemple $\mathbf{S} = [0 \ 0 \ 1]$). L'émergence de ce vecteur non nul certifie mathématiquement au décodeur la présence d'une erreur [4], initiant ainsi la procédure de correction.

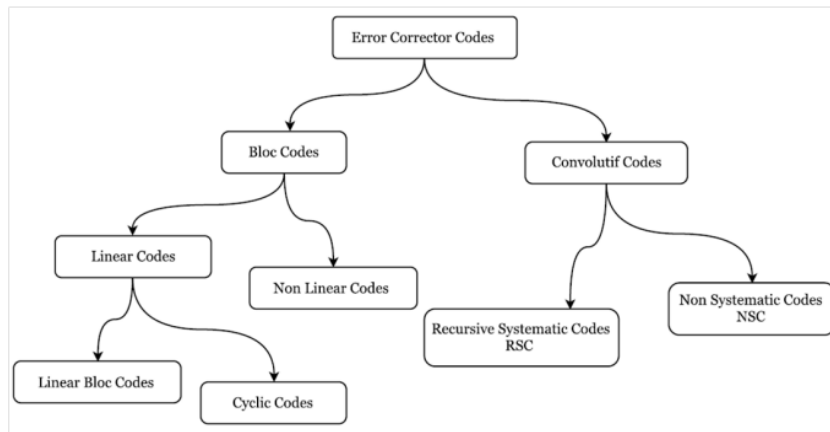


FIGURE 1.2 – Architecture générale d'un système de codage et décodage en bloc linéaire.

1.3.2 Les codes convolutifs

Pour contourner la segmentation rigide imposée par les codes en bloc, un nouveau paradigme a été introduit par Peter Elias en 1955 [6] : les codes convolutifs.

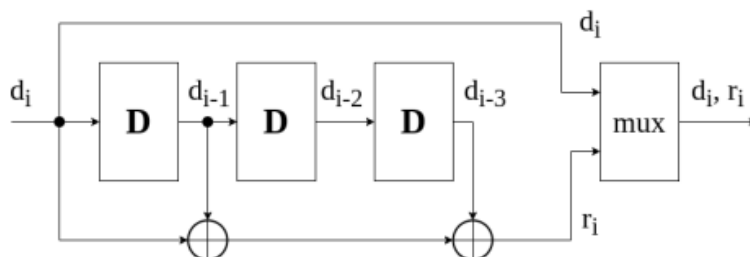


FIGURE 1.3 – Codeur convolutif de fonction de transfert $[1, 1 + D + D^3]$.

Contrairement aux codes en bloc, l'information n'est plus découpée de manière statique. Le flux de données est traité comme une séquence semi-infinie qui traverse une série de registres à décalage [6]. La particularité fondamentale de ces codes est leur "mémoire" : les bits de redondance générés à un instant t dépendent non seulement du bit d'information actuel, mais également des bits précédemment injectés dans le codeur [4].

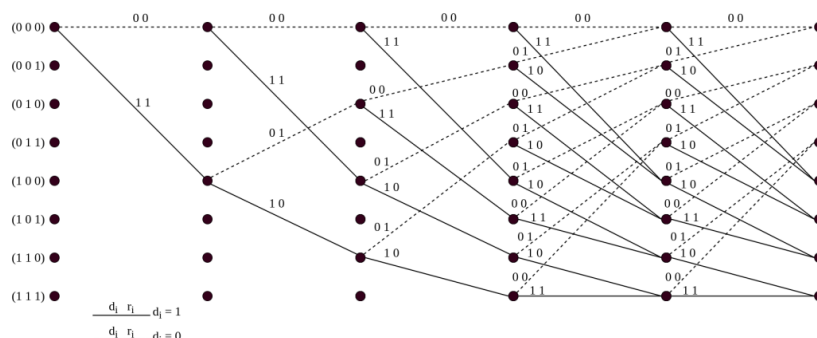


FIGURE 1.4 – Représentation du code convolutif sous forme de diagramme en treillis.

L'évolution des états de ces registres se modélise visuellement par un diagramme en treillis (Figure 1.4). À la réception, le décodage consiste à trouver le chemin le plus probable à travers ce treillis. Cette opération est brillamment réalisée par l'algorithme de Viterbi [6] pour une décision binaire, ou par l'algorithme BCJR pour fournir des informations de fiabilité pondérée [4]. Grâce à leur excellente résistance aux erreurs et à la simplicité de leur implémentation, les codes convolutifs ont dominé les premières générations de communications mobiles (GSM, UMTS).

1.3.3 Les Turbo-codes

L'avènement des Turbo-codes en 1993, sous l'impulsion de Claude Berrou et son équipe [6, 4], a marqué un tournant décisif dans la théorie du codage de canal. Cette famille de codes se distingue par une architecture en concaténation parallèle de deux (ou plusieurs)

codeurs convolutifs récurrents systématiques. Le composant fondamental de cette structure demeure l'entrelaceur : son rôle est de réorganiser de manière pseudo-aléatoire la séquence des bits de données avant leur passage dans le second codeur, permettant ainsi de décorréliser les motifs d'erreurs et de maximiser la distance minimale du code [6].

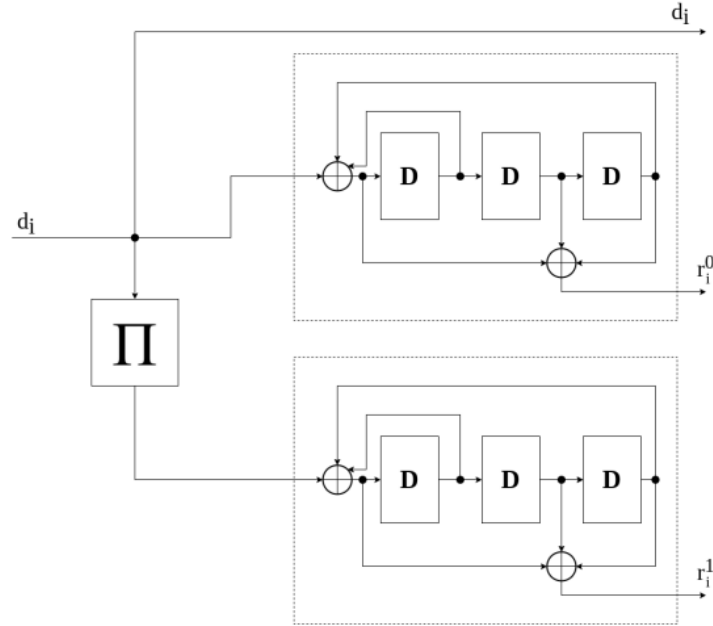


FIGURE 1.5 – Architecture d'un système de Turbo-codage basé sur la concaténation parallèle.

Au niveau du récepteur, le décodage repose sur un principe itératif, tel que détaillé par le schéma de la Figure 1.6. Deux décodeurs à entrées et sorties souples (décodeurs SISO) s'échangent mutuellement des informations probabilistes, dites informations extrinsèques, au fil des itérations. Ce mécanisme de rétroaction permet d'affiner de manière continue la fiabilité des décisions logiques, ce qui permet aux performances du système de converger vers la limite théorique de Shannon [4].

Sur le plan de l'implémentation matérielle, ce traitement séquentiel et itératif génère une latence temporelle significative. Cette contrainte de temps de calcul restreint l'utilisation des Turbo-codes dans les architectures de communication nécessitant des débits de données extrêmement élevés ou des réponses en temps réel strict.

1.3.4 Les codes polaires

Plus récemment, une nouvelle classe de codes a bouleversé la théorie de l'information : les codes polaires, inventés par Erdal Arıkan en 2008 [4].

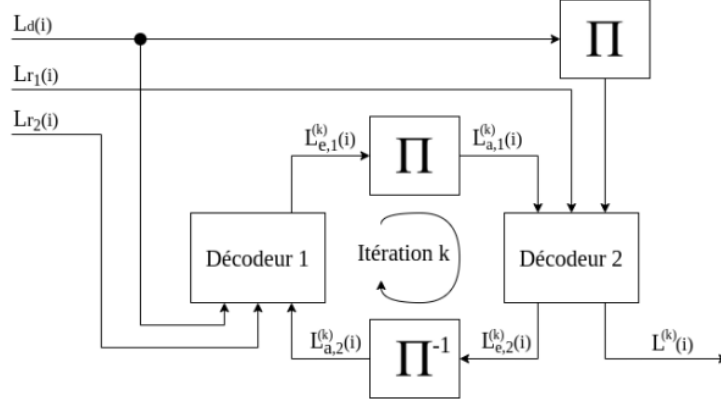


FIGURE 1.6 – Processus de Turbo-décodage itératif avec échange d'informations extrinsèques.

Leur fonctionnement repose sur une découverte mathématique appelée la "polarisation de canal". En combinant intelligemment plusieurs canaux de transmission physiques, l'algorithme crée des canaux "virtuels" aux propriétés extrêmes : certains deviennent parfaitement fiables (sans aucun bruit), tandis que d'autres deviennent totalement illisibles [4]. L'astuce du codage polaire consiste à n'envoyer l'information utile que sur les canaux fiables, et à "geler" les canaux corrompus en y plaçant des bits connus à l'avance (généralement des zéros).

Les codes polaires sont historiquement les premiers codes dont il a été prouvé mathématiquement qu'ils atteignent la capacité théorique d'un canal discret. Leurs performances exceptionnelles, notamment sur les paquets de petite taille, ont conduit à leur adoption immédiate pour sécuriser les canaux de contrôle de la norme 5G [4].

1.3.5 Comparaison des performances

Pour conclure cette synthèse théorique, une évaluation comparative des différentes familles de codes modernes s'impose. Dans ce contexte, les travaux récents menés par Cuc et al. (2023) [7] proposent une analyse rigoureuse des capacités de correction respectives des Turbo-codes, des codes polaires et des codes LDPC. Cette étude a été spécifiquement réalisée sur un canal à Bruit Blanc Gaussien Additif (AWGN) contraint par des interférences, en s'appuyant sur un filtre d'égalisation selon le critère de l'Erreur Quadratique Moyenne Minimale (MMSE).

L'examen des résultats de la Figure 1.7 révèle de manière univoque les excellentes dispositions des codes LDPC. Plus précisément, dès lors que le rapport signal-sur-bruit (SNR) franchit le seuil critique de 1.5 dB, le Taux d'Erreur Binaire (BER) associé au code LDPC amorce une décroissance abrupte, caractéristique de la zone de *waterfall* (chute d'eau). Pour un SNR de 3 dB, cette architecture garantit un taux d'erreur de l'ordre de 10^{-5} , surclassant ainsi significativement les Turbo-codes — qui plafonnent aux alentours de 10^{-4} — ainsi que les codes polaires.

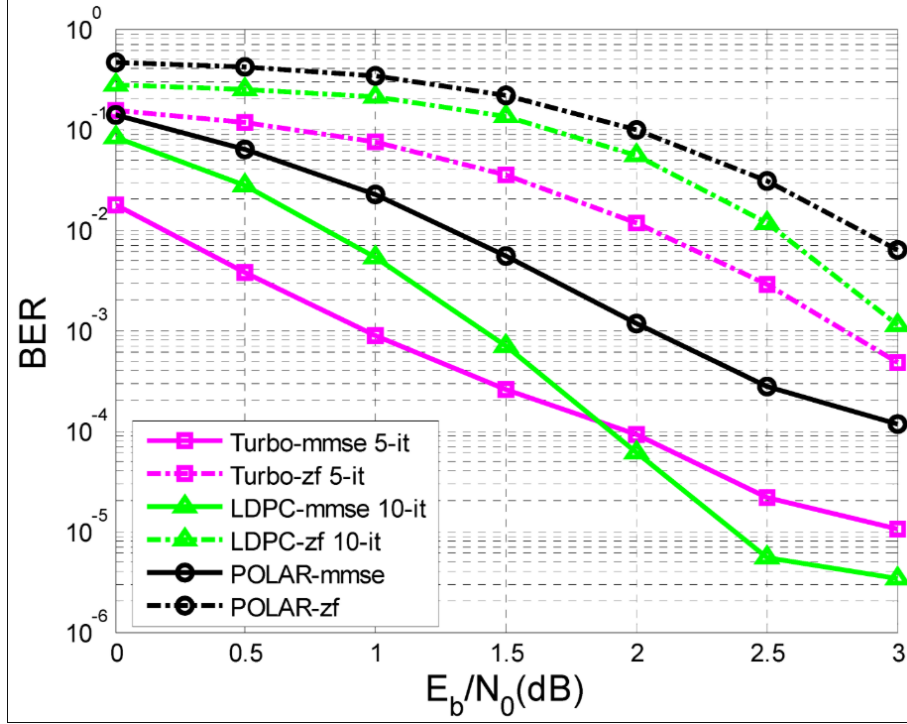


FIGURE 1.7 – Comparaison des performances (BER) entre Turbo-codes, codes LDPC et codes polaires sous égalisation MMSE (Adapté de [7]).

Cette résilience remarquable face aux perturbations du canal physique et aux interférences justifie pleinement le choix de l'architecture LDPC comme socle de notre système. C'est donc exclusivement sur cette technologie, particulièrement adaptée au décodage itératif, que porteront nos travaux d'optimisation algorithmique par Apprentissage Automatique (Machine Learning) développés dans la suite de ce mémoire.

1.4 Conclusion du chapitre

Ce premier chapitre a permis d'établir les fondements théoriques indispensables à la compréhension des systèmes de transmission numérique et du codage de canal. L'évolution des techniques de correction d'erreurs témoigne d'une recherche constante d'optimisation : des premières approches algébriques par blocs (telles que Hamming et BCH) et des structures à mémoire (codes convolutifs), jusqu'à l'introduction des mécanismes itératifs (Turbo-codes) qui ont franchi un palier décisif en s'approchant de la limite de Shannon [6].

Néanmoins, le déploiement des standards de télécommunications de nouvelle génération — à l'image de la 5G NR ou des liaisons satellitaires à très haut débit [4] — impose un cahier des charges beaucoup plus strict. Les systèmes de codage actuels doivent désormais concilier des performances de correction quasi optimales avec une architecture matérielle hautement parallélisable, seule capable de répondre aux exigences de latence ultra-faible.

À la lumière des évaluations de performances récentes [7], les codes LDPC s'imposent comme le compromis technologique le plus viable pour répondre à ces défis, affichant une robustesse face aux interférences qui surclasse celle des Turbo-codes et des codes polaires.

Par conséquent, le chapitre suivant sera entièrement dédié à l'analyse de la structure des codes LDPC et à la définition de la méthodologie de simulation. Après avoir exposé leur formalisme algébrique et topologique à travers le graphe de Tanner, nous analyserons les principes de fonctionnement des algorithmes de décodage classiques à décision stricte, en particulier le *Bit Flipping*. L'étude de leurs limites structurelles face aux configurations de minima locaux permettra de formaliser rigoureusement le besoin d'optimisation, préparant ainsi le cadre théorique nécessaire avant l'introduction de l'approche par Apprentissage Automatique (*Machine Learning*).

Chapitre 2

Structure des codes LDPC

2.1 Introduction

Dans le prolongement des concepts théoriques établis précédemment, ce deuxième chapitre définit le socle expérimental et architectural indispensable à l'implémentation des codes LDPC (Low-Density Parity-Check). Avant d'envisager toute démarche d'optimisation algorithmique, la rigueur scientifique exige en effet d'élaborer une modélisation exacte de la chaîne de transmission classique, permettant ainsi d'évaluer objectivement les performances et les limitations des décodeurs standards.

L'étude débute par une analyse approfondie des caractéristiques structurelles de ces codes. En s'appuyant sur la dualité existant entre leur formalisme matriciel et la topologie du graphe de Tanner introduite au chapitre précédent, nous détaillerons les propriétés algébriques qui régissent leur comportement. Cette base théorique nous permettra ensuite de concevoir un modèle de simulation de bout en bout, englobant toutes les étapes du processus de communication : depuis la génération du flux binaire et sa modulation en bande de base, jusqu'à l'incorporation des modèles statistiques liés aux perturbations du canal physique.

Enfin, ce chapitre se concentrera sur l'évaluation expérimentale de l'algorithme de décodage déterministe à décision stricte GDBF (Gradient Descent Bit Flipping). L'analyse de sa dynamique de convergence et l'extraction rigoureuse de ses métriques — aboutissant à la mise en évidence du phénomène de plancher d'erreur (*error floor*) — constitueront la ligne de base (*baseline*) de référence. C'est l'identification précise des vulnérabilités de cette méthode classique face aux minima locaux qui justifiera la nécessité d'une rupture technologique dans la suite de nos travaux.

2.2 Fondements des codes Low-Density Parity-Check (LDPC)

Les performances de correction des codes LDPC sont intrinsèquement liées à leur définition algébrique, et plus particulièrement à la faible densité de leur matrice de contrôle. Cette contrainte de parcimonie n'est pas qu'une simple particularité mathématique : elle est la condition matérielle qui permet d'exécuter des algorithmes de décodage itératifs avec une complexité de calcul réalisable en pratique.

L'objectif de cette section est de définir rigoureusement cette architecture. Nous analyserons d'abord la construction de la matrice creuse fondamentale, paramètre central du code, avant de transposer ce modèle vers la représentation topologique qui servira de socle à l'implémentation de notre simulateur dans les sections suivantes.

2.2.1 Définition algébrique et historique

Proposés initialement en 1962 par Robert G. Gallager dans le cadre de sa thèse doctorale au Massachusetts Institute of Technology [8], les codes LDPC constituent une classe particulière de codes en bloc linéaires. Bien que leurs capacités théoriques aient été démontrées dès leur introduction, leur implémentation pratique est restée irréalisable pendant plus de trois décennies. Ce frein technologique s'expliquait principalement par la complexité calculatoire prohibitive du processus itératif vis-à-vis des ressources matérielles disponibles à l'époque. Il a fallu attendre le milieu des années 1990 pour assister à la redécouverte de ces codes, notamment grâce aux travaux de MacKay et Neal. Ces derniers ont mis en évidence qu'associés à un décodage itératif (propagation de croyances), les codes LDPC permettaient d'atteindre des performances asymptotiques très proches de la limite théorique de Shannon [9].

D'un point de vue strictement algébrique, un code LDPC est modélisé par le triplet (N, K, d_{min}) , où N correspond à la longueur du mot de code, K définit la dimension du vecteur d'information source, et d_{min} représente la distance minimale de Hamming. Le processus d'encodage consiste à projeter un vecteur d'information $\mathbf{u} \in GF(2)^K$ vers un mot de code valide $\mathbf{v} \in GF(2)^N$ par l'intermédiaire d'une matrice génératrice $\mathbf{G}$ de dimensions KN . Cette transformation s'effectue via la relation linéaire :

$$\mathbf{v} = \mathbf{u} \cdot \mathbf{G} \quad (2.1)$$

Structurellement, l'espace de ce code correspond au noyau d'une matrice d'application linéaire, appelée matrice de contrôle de parité $\mathbf{H}$, de dimensions MN (où $M = N - K$). Par conséquent, pour qu'un vecteur $\mathbf{v}$ soit considéré comme un mot de code valide, il doit impérativement satisfaire l'équation homogène suivante dans le corps de Galois $GF(2)$ [6] :

$$\mathbf{H} \cdot \mathbf{v}^T = \mathbf{0} \pmod{2} \quad (2.2)$$

Cette condition structurelle garantit l'orthogonalité absolue entre les espaces engendrés par l'émission et la réception, ce qui se traduit matriciellement par l'identité fondamentale $\mathbf{G} \cdot \mathbf{H}^T = \mathbf{0}$.

2.2.2 La matrice de contrôle de parité $\mathbf{H}$ et classification

L'identité fondamentale des codes LDPC repose sur les propriétés algébriques de leur matrice de contrôle de parité $\mathbf{H}$, et plus spécifiquement sur sa nature creuse (*sparse matrix*). À l'inverse des matrices denses associées aux codes en bloc traditionnels (tels que les codes BCH), la densité des éléments non nuls (les '1') au sein de la matrice $\mathbf{H}$ est extrêmement faible et tend asymptotiquement vers zéro à mesure que la longueur du bloc N s'accroît [4]. Cette parcimonie matricielle constitue le paramètre déterminant pour la viabilité de l'implémentation. En limitant le nombre d'interdépendances logiques lors de l'évaluation des syndromes, elle maintient la complexité calculatoire des algorithmes itératifs à un niveau matériellement réalisable [6].

En fonction des règles d'agencement de ces éléments non nuls, l'architecture des codes LDPC se divise en deux catégories structurelles :

- **Les codes LDPC réguliers :** La matrice $\mathbf{H}$ est qualifiée de régulière si les poids de Hamming de ses vecteurs colonnes et lignes sont invariants. Concrètement, chaque colonne comporte exactement d_v éléments non nuls (degré des variables) et chaque ligne en comporte d_c (degré des contrôles), sous réserve que $d_v \ll M$ et $d_c \ll N$ pour respecter le critère de faible densité [8]. Dans cette configuration homogène, la borne inférieure du rendement du code R est déduite analytiquement par l'expression :

$$R \geq 1 - \frac{d_v}{d_c} \quad (2.3)$$

- **Les codes LDPC irréguliers :** Lorsque le nombre de '1' varie d'une colonne à l'autre ou d'une ligne à l'autre, le code adopte une topologie irrégulière. Ces fluctuations ne sont pas aléatoires ; elles sont rigoureusement modélisées par des polynômes de distribution de degrés. On définit $\lambda(x)$ pour les colonnes et $\rho(x)$ pour les lignes selon les relations [9] :

$$\lambda(x) = \sum_{i=2}^{d_v^{max}} \lambda_i x^{i-1} \quad \text{et} \quad \rho(x) = \sum_{j=2}^{d_c^{max}} \rho_j x^{j-1} \quad (2.4)$$

où les coefficients λ_i et ρ_j quantifient la fraction des arêtes globalement connectées aux nœuds de degré i et j . Bien que l'optimisation mathématique de cette hétérogénéité augmente considérablement la complexité de conception de la matrice, elle procure un avantage décisif au système : elle améliore la dynamique de convergence du décodeur, surclassant ainsi les structures régulières pour frôler les

limites théoriques de capacité du canal.

2.2.3 Représentation topologique : Le graphe de Tanner

Au-delà de sa modélisation purement matricielle, un code LDPC peut être décrit par une représentation topologique introduite par R. Tanner en 1981 [10]. Cette approche modélise le code sous la forme d'un graphe biparti. Ce formalisme visuel revêt une importance capitale en ingénierie, car il définit l'architecture physique de routage sur laquelle s'exécutent les algorithmes de décodage itératif.

La nature bipartite du graphe implique une stricte dichotomie des nœuds en deux ensembles distincts, interdisant formellement toute interconnexion entre les éléments d'un même groupe :

- **Les nœuds de variable (VN - *Variable Nodes*)** : Identifiés par V_j (avec $j \in \{1, 2, \dots, N\}$), ils sont directement associés aux colonnes de la matrice $\mathbf{H}$ et matérialisent les différents bits constituant le mot de code reçu.
- **Les nœuds de contrôle (CN - *Check Nodes*)** : Notés C_i (avec $i \in \{1, 2, \dots, M\}$), ils correspondent aux vecteurs lignes de la matrice $\mathbf{H}$ et incarnent les contraintes de parité que le système global doit satisfaire.

La règle de routage du graphe est directement dictée par les coefficients de la matrice : une arête physique connecte un nœud de variable V_j à un nœud de contrôle C_i si et seulement si l'élément d'intersection $H_{i,j}$ est non nul ($H_{i,j} = 1$). Pour concrétiser cette dualité algèbro-topologique, prenons l'exemple d'un code défini par une matrice $\mathbf{H}$ de dimensions 69 :

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.5)$$

L'architecture de ce code se traduit par un treillis comportant 9 nœuds variables et 6 nœuds de contrôle. Si l'on examine la quatrième colonne, les éléments non nuls se situent aux lignes 1, 4 et 6. Sur le plan matériel, cela indique que le nœud V_4 présente un degré de connectivité de 3, le raccordant par des arêtes aux nœuds C_1 , C_4 et C_6 . Réciproquement, la deuxième ligne de la matrice impose des liens avec les colonnes 2, 3, 5 et 8. La contrainte d'intégrité gérée par le nœud C_2 s'exprime alors rigoureusement par la somme modulo 2 suivante : $v_2 \oplus v_3 \oplus v_5 \oplus v_8 = 0$.

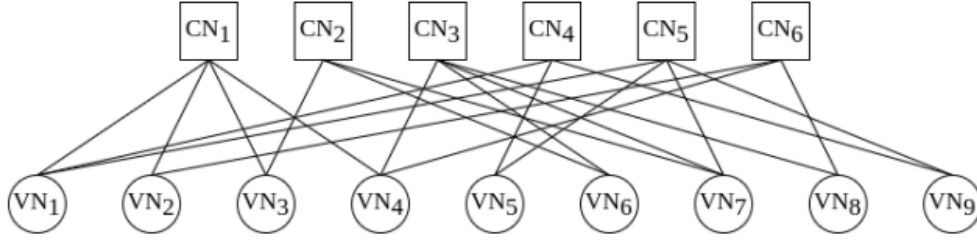


FIGURE 2.1 – Représentation topologique d’un code LDPC via un graphe bipartite de Tanner mettant en évidence le routage entre nœuds de variable (VN) et nœuds de contrôle (CN).

Comme illustré par la Figure 2.1, la stricte séparation bipartite est une propriété fondamentale. L’absence de connexions intra-groupe garantit que, lors de la phase de décodage, l’information probabiliste traitée par un nœud donné provient exclusivement de sources indépendantes situées dans le groupe opposé. Ce principe de non-corrélation locale est la condition *sine qua non* permettant la convergence des algorithmes de décodage par échange de messages (*Message-Passing algorithms*).

2.2.4 Les codes LDPC Quasi-Cycliques (QC-LDPC)

L’implémentation matérielle des codes LDPC basés sur des matrices de parité totalement aléatoires se heurte à des verrous technologiques majeurs lors des phases de synthèse logique et de routage sur les circuits intégrés ou les architectures de type FPGA. L’absence de structure géométrique au sein de la matrice implique une complexité d’interconnexion extrême, conduisant à une congestion critique des pistes physiques sur le silicium. De plus, le stockage des positions des éléments non nuls requiert une allocation prohibitive de ressources mémoires, limitant la viabilité pratique du dispositif de décodage [4].

Pour concilier les performances de correction d’erreurs et les contraintes de faisabilité industrielle, les standards de télécommunications à très haut débit — tels que la 5G New Radio (NR), le Wi-Fi 6 (IEEE 802.11ax) ou les transmissions satellitaires de nouvelle génération — exploitent de manière privilégiée une sous-famille structurée : les codes LDPC Quasi-Cycliques (QC-LDPC).

Dans l’architecture d’un code QC-LDPC, la matrice globale de contrôle de parité $\mathbf{H}$ abandonne la distribution arbitraire des éléments binaires isolés au profit d’un agencement modulaire partitionné en blocs [4]. Elle est structurée sous la forme d’une grille de macro-matrices composées de sous-matrices carrées de dimensions $Z \times Z$, où le paramètre entier Z définit le facteur d’expansion (*lifting size*). Chacun de ces sous-blocs élémentaires, noté $\mathbf{P}^{p,q}$, répond à l’une des deux configurations exclusives suivantes :

- **Une matrice nulle (0) :** Un bloc carré d’ordre Z exclusivement composé de zéros,

traduisant l'absence de connexions topologiques entre les macro-nœuds associés.

- **Une matrice circulante de permutation :** Une matrice identité d'ordre Z ayant subi un décalage circulaire vers la droite d'un nombre entier de positions, spécifié par le coefficient de décalage $S_{p,q}$ (avec $0 \leq S_{p,q} < Z$).

À titre d'exemple, pour un facteur d'expansion fixé à $Z = 4$ et un décalage unitaire $S_{p,q} = 1$, le sous-bloc circulant prend la forme algébrique suivante :

$$\mathbf{P}^1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad (2.6)$$

Cette organisation permet de compacter la description du code au moyen d'une matrice de base abrégée, notée $\mathbf{H}_{base}$. Dans ce formalisme condensé, chaque sous-bloc est uniquement caractérisé par son coefficient de décalage $S_{p,q}$, tandis que la valeur -1 est affectée aux blocs entièrement nuls.

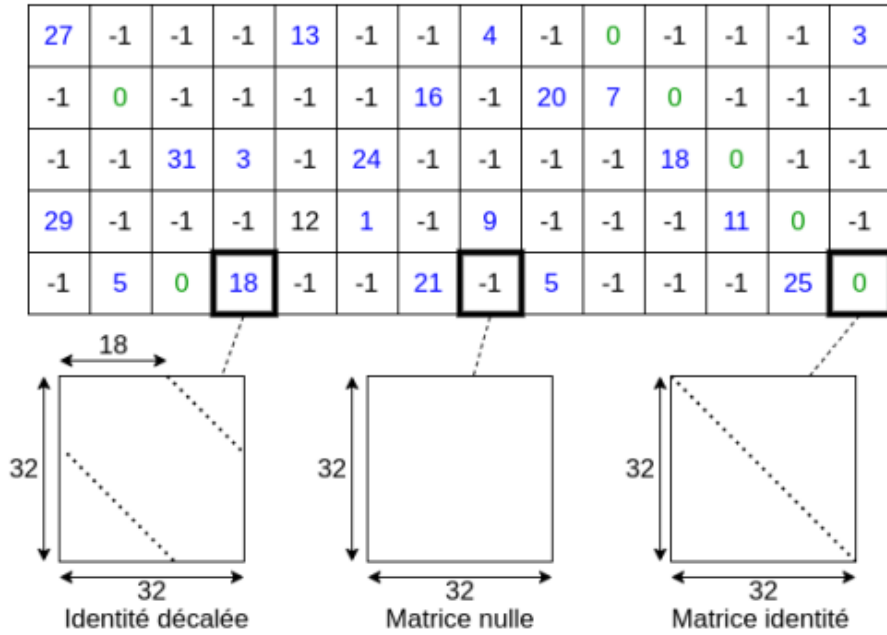


FIGURE 2.2 – Structure macro-matricielle d'un code QC-LDPC de dimensions 448160 pour un facteur d'expansion $Z = 32$.

L'ordonnancement régulier de ces sous-matrices circulantes est illustré par la Figure 2.2 pour une configuration de dimensions 448160 avec un facteur d'expansion $Z = 32$. Du point de vue de l'ingénierie des circuits, cette régularité géométrique est capitale. Elle

supprime le besoin de stocker l'intégralité de la matrice en mémoire RAM, la mémorisation de la matrice de base $\mathbf{H}_{base}$ et des coefficients de décalage suffisant à caractériser le code complet. De surcroît, elle simplifie radicalement les architectures d'accès mémoire en permettant de paralléliser massivement le traitement de Z nœuds simultanément sans générer de conflits sur les bus de données [4].

2.3 Algorithmes de décodage itératifs classiques

Le recouvrement du mot de code original, à partir des séquences altérées par le canal, s'effectue via des algorithmes itératifs d'échange de messages. Ces processus de résolution s'appuient strictement sur les contraintes algébriques définies par la matrice de contrôle de parité $\mathbf{H}$. En fonction du format de l'information délivrée par l'étage de démodulation et manipulée par le décodeur, la littérature distingue deux paradigmes fondamentaux : le décodage à décision stricte (*Hard-Decision*) et le décodage à décision pondérée (*Soft-Decision*).

2.3.1 Le décodage à décision stricte (Hard-Decision)

Dans l'architecture à décision stricte, le signal analogique issu du canal subit une quantification binaire immédiate (sur un seul bit) préalablement à son traitement par le décodeur. L'atout majeur de cette approche réside dans sa compacité matérielle et la faible complexité de son implémentation, la charge de calcul se résumant à des opérations logiques booléennes élémentaires [6].

Cependant, cette simplification du signal entraîne une perte d'information définitive. En forçant la projection du signal sur un bit net, le récepteur s'ampute de la mesure de "fiabilité" intrinsèque au canal physique. À titre d'exemple, un symbole reçu avec une forte amplitude énergétique et un symbole très ambigu, frôlant le seuil critique de décision, seront évalués avec un niveau de confiance rigoureusement identique par le décodeur. L'algorithme de *Bit Flipping* (BF), que nous détaillerons dans la section subséquente, constitue le standard de référence de cette catégorie pour l'évaluation des codes LDPC [8].

2.3.2 Le décodage à décision pondérée (Soft-Decision)

En opposition aux méthodes strictes, le décodage à décision pondérée préserve et exploite la dimension probabiliste continue du canal de transmission. Le décodeur ne manipule plus de simples variables binaires, mais traite des grandeurs réelles qui quantifient finement la fiabilité de chaque observation [4].

Pour formaliser mathématiquement cette incertitude, la métrique universellement retenue est le **Rapport de Vraisemblance Logarithmique** (*Log-Likelihood Ratio* ou **LLR**). Pour un bit codé émis c_i soumis à une observation y_i en réception, le LLR s'exprime par la relation :

$$L(c_i) = \ln \left(\frac{P(c_i = 0 \mid y_i)}{P(c_i = 1 \mid y_i)} \right) \quad (2.7)$$

Cette grandeur condense deux informations vitales pour la dynamique itérative : le signe algébrique du LLR indique l'état logique le plus probable (par convention, une valeur positive pointe vers un '0', une valeur négative vers un '1'), tandis que son amplitude absolue $|L(c_i)|$ mesure le degré de certitude associé à cette estimation. Une amplitude tendant vers zéro caractérise ainsi une indécision prononcée, symptomatique d'une forte corruption du signal par le bruit additif du canal [4].

2.4 L'algorithme Sum-Product (SPA) / Belief Propagation

Dans un décodeur à décision pondérée, le récepteur ne quantifie pas brutalement le signal reçu en 0 ou 1. Il conserve et exploite une information de fiabilité issue du démodulateur, généralement représentée sous la forme d'un rapport de vraisemblance logarithmique (LLR - *Log-Likelihood Ratio*). Cette valeur réelle indique la probabilité qu'un bit transmis soit un '0' ou un '1' [6]. L'algorithme de référence dans cette catégorie est l'algorithme

Sum-Product (SPA), souvent appelé *Belief Propagation* (BP). Son principe fondamental repose sur l'échange itératif et continu d'informations extrinsèques (des messages de probabilités) entre les nœuds de variable et les nœuds de contrôle directement sur le graphe de Tanner [4]. Ce processus, qui vise à affiner la certitude sur la valeur de chaque bit, s'articule autour des trois étapes suivantes :

1. **Initialisation** : Chaque nœud de variable est initialisé avec la valeur du LLR issue du canal.
2. **Mise à jour itérative** :
 - **Nœuds de variable vers nœuds de contrôle** : Chaque nœud de variable transmet à ses nœuds de contrôle voisins la probabilité qu'il vaille 0 ou 1, en se basant sur les messages reçus de l'itération précédente (à l'exclusion de l'information provenant du nœud destinataire).
 - **Nœuds de contrôle vers nœuds de variable** : Chaque nœud de contrôle évalue la probabilité que son équation de parité soit satisfaite et renvoie cette information aux nœuds de variable connectés.

3. **Décision finale** : L'itération s'arrête lorsque toutes les équations de parité sont vérifiées (le syndrome est nul) ou lorsqu'un nombre maximum d'itérations est atteint. La décision finale est prise en évaluant le signe du LLR *a posteriori* global [4].

Voici la Figure 2.3 qui illustre parfaitement l'organigramme et le fonctionnement de cet algorithme *Sum-Product* :

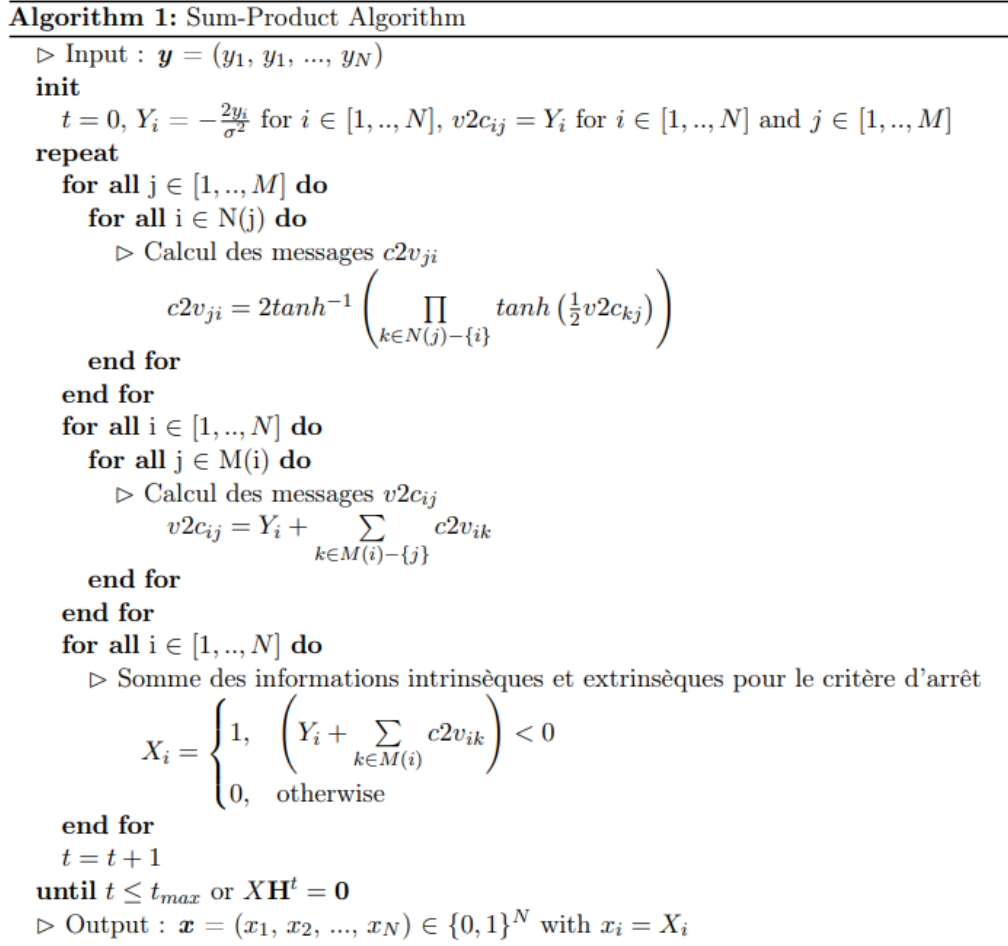


FIGURE 2.3 – Organigramme de l'algorithme de décodage Sum-Product (BP) [6].

Bien que le décodage *Soft-Decision* offre d'excellentes performances qui frôlent la limite de Shannon, il présente une complexité calculatoire prohibitive. En effet, l'évaluation des messages probabilistes nécessite l'utilisation intensive de fonctions non-linéaires (telles que les tangentes hyperboliques) et de nombreuses multiplications complexes en virgule flottante [6, 4]. Cette lourdeur algorithmique pose de sérieux défis pour l'implémentation matérielle, limitant la montée en débit et augmentant drastiquement la consommation énergétique.

2.5 L'Algorithme Bit Flipping (BF)

C'est précisément pour contourner la complexité matérielle écrasante des algorithmes à décision pondérée (comme le SPA) que les décodeurs à décision stricte (*Hard-Decision*) s'imposent comme une alternative redoutable. Dans cette approche, le signal analogique reçu du canal est immédiatement quantifié sur un seul bit (0 ou 1). L'algorithme le plus emblématique de cette famille est le *Bit Flipping* (BF), introduit originellement par Gallager [8].

Le principe fondamental du *Bit Flipping* repose sur une logique booléenne simple et itérative. À chaque itération, le décodeur calcule le syndrome binaire du mot reçu pour identifier les bits qui violent le plus grand nombre d'équations de parité. Le déroulement précis de cette méthode, illustré par l'organigramme de la Figure 2.4, s'articule autour des étapes suivantes :

```

1: procedure DECODE(y)
2:
3:    $I = 0$  ▷ Initialization
4:   for  $i = 1 : n$  do
5:      $M_i = y_i$ 
6:   end for
7:
8:   repeat
9:     for  $j = 1 : m$  do ▷ Step 1: Check messages
10:      for  $i = 1 : n$  do
11:         $E_{j,i} = \sum_{i' \in B_j, i' \neq i} (M_{i'} \bmod 2)$ 
12:      end for
13:    end for
14:
15:    for  $i = 1 : n$  do ▷ Step 2: Bit messages
16:      if the messages  $E_{j,i}$  disagree with  $y_i$  then
17:         $M_i = (r_i + 1 \bmod 2)$ 
18:      end if
19:    end for
20:
21:    for  $j = 1 : m$  do ▷ Test: are the parity-check
22:       $L_j = \sum_{i' \in B_j} (M_{i'} \bmod 2)$  ▷ equations satisfied
23:    end for
24:    if all  $L_j = 0$  or  $I = I_{\max}$  then
25:      Finished
26:    else
27:       $I = I + 1$ 
28:    end if
29:  until Finished
30: end procedure

```

FIGURE 2.4 – Procédure de l'algorithme de décodage Bit Flipping [6].

1. **Initialisation** : Le signal analogique reçu $\mathbf{y}$ est quantifié (Hard Decision) pour

former le mot binaire initial $\mathbf{x} = (x_1, x_2, \dots, x_n)$ [6].

2. **Vérification du syndrome** : Le décodeur évalue l'état de chaque équation de parité. Si toutes les équations sont satisfaites (syndrome nul), l'algorithme s'arrête et génère le mot décodé [6].
3. **Évaluation (Fonction d'inversion)** : Pour chaque nœud de variable x_k , le décodeur calcule une "fonction d'inversion" $\Delta_k(\mathbf{x})$, qui agit comme une mesure d'invalidité de l'affectation du bit [4]. Dans le BF classique, cela correspond au nombre exact de nœuds de contrôle insatisfaits connectés à x_k .
4. **Inversion (Flipping)** : Le décodeur identifie la position l qui minimise (ou maximise, selon la convention) la fonction d'inversion, et inverse le bit correspondant ($x_l := -x_l$ en notation bipolaire) [6].
5. **Itération** : Le processus retourne à l'étape 2 jusqu'à ce que le syndrome soit nul ou qu'un nombre maximum d'itérations L_{max} soit atteint [6, 4].

Avantages matériels et limites théoriques : D'un point de vue de l'ingénierie matérielle, l'algorithme *Bit Flipping* est extrêmement séduisant. En ne manipulant que des valeurs binaires, il élimine totalement le besoin d'unités de calcul en virgule flottante. Son implémentation sur silicium (FPGA/ASIC) se contente de portes logiques basiques (XOR, AND) et de simples compteurs [4]. Cette légèreté architecturale permet une parallélisation massive et garantit des débits ultra-élevés avec une très faible consommation énergétique.

Toutefois, cette simplification extrême a un coût théorique inévitable. La quantification immédiate du signal détruit l'information précieuse de fiabilité du canal, ce qui dégrade drastiquement le pouvoir de correction du décodeur face à un bruit important [6].

2.6 Comparaison des décodeurs et justification du choix

Afin de synthétiser les différences fondamentales entre ces deux paradigmes de décodage (décision pondérée et décision stricte), le Tableau 2.1 met en opposition leurs caractéristiques algorithmiques et matérielles.

L'interprétation de ce tableau est sans appel : le concepteur d'un système de télécommunication fait face à un compromis strict entre la puissance de correction d'erreurs (SPA) et la simplicité d'intégration matérielle (BF) [6, 4]. L'enjeu scientifique de ce projet de fin d'études est de briser ce compromis traditionnel. Il s'agira, dans la suite de ce mémoire, de conserver la légèreté architecturale de la décision stricte tout en y injectant l'intelligence de l'*Apprentissage Automatique (Machine Learning)*. Cette approche novatrice permettra

Caractéristique	Sum-Product Algorithm (Soft-Decision)	Bit Flipping (Hard-Decision)
Type de données manipulées	Valeurs réelles (Probabilités / LLR)	Valeurs binaires (0 ou 1)
Performances (BER / Gain)	Excellentes (proches de la limite de Shannon)	Médiocres (perte significative d'information)
Opérations mathématiques	Additions, multiplications, tangentes hyperboliques	Opérations logiques simples (XOR, AND, compteurs)
Complexité (Hardware)	Très élevée (grande surface silicium, routage dense)	Très faible (architecture simple et compacte)
Latence et débit	Latence élevée, débit modéré	Faible latence, très haut débit (adapté au temps réel)

TABLE 2.1 – Comparaison entre le décodage à décision pondérée (SPA) et stricte (BF)

de piloter dynamiquement le *Bit Flipping* pour atteindre des performances de correction rivalisant avec celles du *Soft-Decision*.

2.7 Le Gradient Descent Bit Flipping (GDBF)

Bien que l'algorithme Bit Flipping (BF) classique se distingue par sa rapidité et la simplicité de son implémentation matérielle, ses capacités de correction demeurent significativement inférieures à celles des algorithmes probabilistes (SPA/Min-Sum). Afin de combler cette fracture de performance, Wadayama et al. ont introduit en 2010 une reformulation mathématique majeure de l'approche à décision stricte [11]. Plutôt que de baser la décision de basculement sur un simple comptage d'équations de parité erronées, le processus de décodage est modélisé de manière rigoureuse comme un problème d'optimisation visant à maximiser une fonction objective globale.

En adoptant une représentation algébrique bipolaire où les états logiques des bits s'expriment par $x_k, y_k \in \{+1, -1\}$, la fonction objective non linéaire $f(\mathbf{x})$ définissant l'état du décodeur est formulée par l'équation suivante [11] :

$$f(\mathbf{x}) = \sum_{k=1}^n x_k y_k + \sum_{i=1}^m \prod_{j \in N(i)} x_j \quad (2.8)$$

L'élégance de cette équation réside dans sa bipartition en deux contraintes physiques distinctes :

- **Le terme de corrélation croisée** ($\sum x_k y_k$) : Il quantifie la fidélité du mot de code candidat $\mathbf{x}$ par rapport aux observations brutes extraites du canal $\mathbf{y}$. Maximiser

ce terme revient à minimiser la distance euclidienne entre la séquence estimée et le signal reçu [11].

- **Le terme de pénalité de syndrome** ($\sum \prod x_j$) : Il évalue la validité topologique sur le graphe de Tanner. Ce terme n'atteint son extremum global que si le vecteur $\mathbf{x}$ satisfait l'intégralité des équations de contrôle de parité, certifiant l'appartenance de $\mathbf{x}$ à l'espace du code [11].

Pour résoudre ce problème d'optimisation multidimensionnelle tout en préservant une complexité matérielle minimale, l'algorithme calcule le gradient local. En dérivant partiellement la fonction objective par rapport à chaque bit, on obtient la fonction d'évaluation (ou d'inversion) spécifique au GDBF [11] :

$$\Delta_k^{(GD)}(\mathbf{x}) = x_k y_k + \sum_{i \in M(k)} \prod_{j \in N(i)} x_j \quad (2.9)$$

Chaque nœud variable calcule ainsi sa propre "énergie" locale, réalisant un compromis instantané entre la fiabilité de son observation canal et l'intégrité de ses équations de parité environnantes.

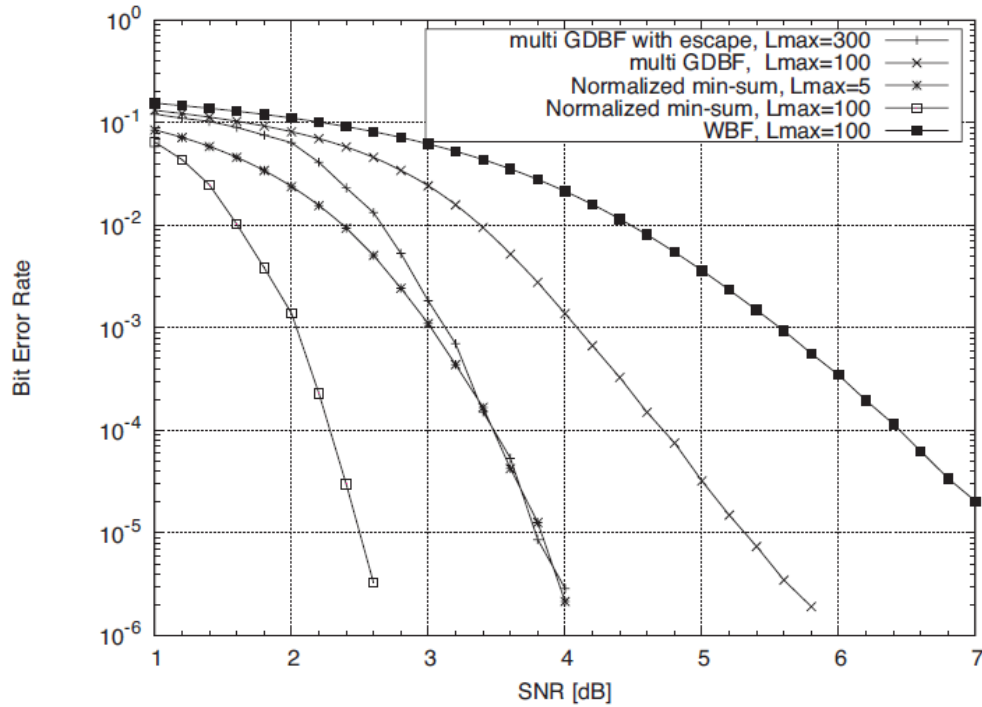


FIGURE 2.5 – Évaluation comparative des performances du GDBF face aux algorithmes heuristiques classiques (WBF, MWBF) sur canal AWGN (Tiré de Wadayama et al., 2010 [11]).

L'analyse de la Figure 2.5 valide expérimentalement la pertinence de cette formulation par gradient. On observe que les variantes du GDBF, et tout particulièrement l'architecture *Multi-GDBF* (autorisant le basculement simultané de plusieurs bits par itération), surclassent nettement les algorithmes de la génération précédente tels que le WBF (*Weighted Bit Flipping*) et le MWBF. La topologie de la courbe démontre une nette accélération dans la zone de *waterfall*. Néanmoins, les résultats soulignent également l'existence d'un plafond de performance résiduel par rapport aux algorithmes à décision pondérée (Min-Sum). Ce blocage n'est pas lié à la puissance de calcul, mais à la topologie de la fonction objective : le paysage d'optimisation étant hautement non convexe, la descente de gradient converge inévitablement vers des maxima locaux, piégeant le décodeur dans un état d'erreur persistant [11].

Pour briser cette stagnation algorithmique, l'équipe de Wadayama a conceptualisé un mécanisme de perturbation dynamique désigné sous le terme de "processus d'échappement" (*escape process*).

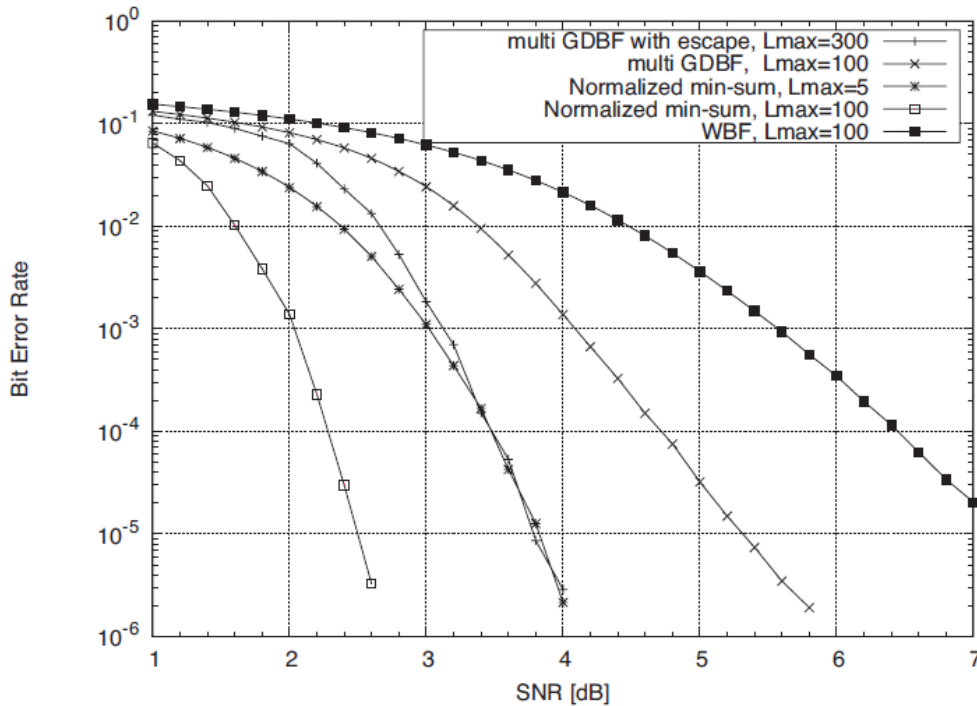


FIGURE 2.6 – Impact du processus d'échappement sur la dynamique de convergence et la mitigation des minima locaux dans l'algorithme GDBF (Tiré de Wadayama et al., 2010 [11]).

La Figure 2.6 illustre de manière spectaculaire le bénéfice de cette injection d'instabilité. En forçant l'algorithme à modifier transitoirement son seuil d'inversion pour s'extraire de l'attraction d'un minimum local, le GDBF avec processus d'échappement enregistre un gain de codage impressionnant d'environ 1.5 dB pour un Taux d'Erreur Binaire (BER)

de 10^{-5} [11].

Bien que le processus d'échappement (*escape process*) introduit par Wadayama et al. valide la nécessité d'une perturbation pour extraire l'algorithme des minima locaux [1], cette méthode présente une limitation algorithmique stricte. Elle repose sur des seuils d'activation empiriques et statiques [1]. Le dimensionnement manuel de ces heuristiques figées est complexe et empêche le décodeur de s'adapter aux variations dynamiques d'un canal de transmission réel [1]. Cette rigidité paramétrique rend l'approche classique sous-optimale et justifie techniquement le recours à l'Apprentissage Automatique (*Machine Learning*). L'intégration de modèles adaptatifs pour optimiser dynamiquement la fonction de décision fait l'objet du chapitre suivant.

2.8 Problématique et Transition vers le Machine Learning

L'analyse des travaux de Wadayama et al. [11] met en évidence une limitation algorithmique stricte du GDBF : le décodeur s'enferme inévitablement dans des minima locaux induits par les ensembles pièges (*Trapping Sets*). Bien que le mécanisme d'échappement théorique prouve la nécessité d'une perturbation, son implémentation classique repose sur des seuils d'activation statiques et empiriques. Cette rigidité paramétrique empêche l'algorithme de s'adapter aux variations dynamiques d'un canal de transmission réel, rendant ces heuristiques figées sous-optimales.

De ce constat technique émerge la problématique centrale qui pilote l'intégralité de ce projet de fin d'études :

Comment optimiser un algorithme de décodage à décision stricte par l'apprentissage automatique pour contourner ses blocages structurels, afin d'approcher les performances des algorithmes complexes sans augmenter l'empreinte matérielle du circuit ?

Pour résoudre cette limitation, il est nécessaire de remplacer les équations de seuil rigides par un modèle adaptatif. L'intégration de l'Apprentissage Automatique (*Machine Learning*) trouve ici sa pleine justification. Plutôt que d'appliquer une règle mathématique de basculement universelle — inefficace sur des topologies complexes —, l'objectif est d'utiliser un apprentissage hors-ligne pour modéliser la signature statistique des erreurs directement à partir du graphe de Tanner [12, 13].

Concrètement, en dotant chaque nœud variable d'une Machine à États Finis (FSM), le modèle agit comme un contrôleur dynamique. Il exploite l'historique local du syndrome pour ajuster l'inertie des bits et forcer l'échappement hors des ensembles pièges. Cette

méthode garantit le déblocage du décodeur par des opérations logiques simples, évitant ainsi le recours aux lourds calculs en virgule flottante exigés par l'algorithme *Sum-Product*.

Ce constat clôture notre étude théorique des architectures de décodage classiques. Les chapitres subséquents constitueront le cœur contributif de ce mémoire. Ils seront intégralement dédiés à la conception, la modélisation sous MATLAB, et l'évaluation expérimentale de cette architecture hybride, où nous détaillerons comment l'optimisation des paramètres de la machine à états redéfinit la dynamique de correction des codes LDPC.

2.9 Conclusion du chapitre

Ce deuxième chapitre a permis d'établir de manière rigoureuse les fondements théoriques, algébriques et topologiques des codes LDPC. L'étude détaillée des différentes familles d'algorithmes de décodage a conduit aux constats analytiques suivants :

- **Structuration topologique** : L'analyse de la matrice de contrôle de parité et de sa représentation sous forme de graphe de Tanner confirme la pertinence des structures Quasi-Cycliques (QC-LDPC) pour garantir une implémentation matérielle viable.
- **Compromis de décodage** : La comparaison des décodeurs a mis en évidence un antagonisme technologique fort entre la décision pondérée (algorithme *Sum-Product*), offrant d'excellentes performances au prix d'une complexité matérielle écrasante, et la décision stricte (*Bit Flipping*), extrêmement légère mais pénalisée par son pouvoir de correction.
- **Limites théoriques du GDBF** : L'analyse de l'algorithme *Gradient Descent Bit Flipping* confirme son fort potentiel pour réduire la complexité d'intégration matérielle. Cependant, l'étude démontre que sa nature purement déterministe l'enferme inévitablement dans des minima locaux causés par les ensembles pièges (*Trapping Sets*), générant ainsi un plancher d'erreur structurel.

Pour surmonter cet obstacle de stagnation sans alourdir la complexité de l'algorithme, une évolution de l'architecture de décodage s'impose. Le chapitre suivant introduira les techniques d'Apprentissage Automatique (*Machine Learning*). Nous y détaillerons comment l'intégration d'une Machine à États Finis (FSM) couplée au GDBF permet de doter le décodeur d'une capacité d'échappement dynamique, le rendant capable de s'extraire de ces impasses topologiques avec une précision purement déterministe.

Chapitre 3

Décodage GDBF à Machine à États Finis : Architecture et Optimisation des Paramètres

3.1 Introduction

Le chapitre précédent a mis en évidence le principal verrou technologique de l'algorithme *Gradient Descent Bit Flipping* (GDBF) [11] : sa vulnérabilité face aux minima locaux. La topologie du graphe de Tanner favorise l'apparition d'ensembles pièges (*Trapping Sets*) qui bloquent le processus itératif et provoquent un plancher d'erreur inacceptable [4, 11].

Historiquement, la littérature a pallié ce blocage par l'injection de perturbations aléatoires (PGDBF) [14, ?]. Toutefois, le déploiement matériel de générateurs de nombres aléatoires augmente drastiquement la surface de silicium et la consommation énergétique, s'éloignant des contraintes des systèmes de télécommunications embarqués.

Ce troisième chapitre propose donc d'explorer une voie technologique novatrice : l'intégration de l'Apprentissage Automatique (*Machine Learning*). L'objectif est de concevoir une architecture adaptative capable d'apprendre la dynamique des erreurs pour s'extraire des impasses topologiques, tout en conservant une logique purement déterministe adaptée aux contraintes d'intégration matérielle.

3.2 État de l'art du décodage Bit-Flipping par Apprentissage Automatique

Pour modéliser statistiquement la dynamique des erreurs et éviter l'utilisation de l'aléatoire, plusieurs paradigmes d'apprentissage automatique ont récemment été appliqués au processus de décodage.

L'optimisation par réseaux de neurones profonds constitue la première approche majeure. Des travaux précurseurs (Nachmani et al. [?], Lugosch et al. [?]) ont démontré la possibilité de "déplier" le graphe de Tanner pour apprendre les poids des messages échangés. Cette méthode supervisée a ensuite été adaptée aux codes LDPC de la norme 5G (Wu et al. [15]) et affinée par l'intégration de poids apprenables (*Learnable Weights*) sur des architectures à décision pondérée [13].

Une seconde approche exploite l'Apprentissage par Renforcement (*Reinforcement Learning*). Carpi et al. [16] ont modélisé le décodage comme un Processus de Décision Markovien (MDP). Dans ce cadre, un agent autonome apprend hors-ligne une politique d'inversion en maximisant la réduction du syndrome, élaborant ainsi des stratégies complexes pour contourner les ensembles pièges.

Cependant, bien que ces techniques maximisent le pouvoir de correction, leur transfert vers une implémentation matérielle (FPGA ou ASIC) pose un problème majeur de

complexité. L'exécution de réseaux de neurones ou de politiques de renforcement exige des calculs en virgule flottante et une empreinte mémoire totalement incompatibles avec les débits de plusieurs Gbit/s exigés par les standards modernes.

Ces spécifications matérielles strictes imposent le recours à une architecture hybride à très faible complexité. C'est précisément pour répondre à ce cahier des charges que s'insère l'utilisation de la Machine à États Finis (FSM) [17]. Cette solution exploite l'intelligence de l'apprentissage hors-ligne pour configurer ses paramètres, tout en conservant à l'exécution une logique combinatoire purement déterministe et compacte.

3.3 Évolution historique et genèse de l'algorithme GDBF-FSM

L'algorithme GDBF originel [11] évalue la fonction d'énergie de manière statique. Cette absence de mémoire provoque un blocage inévitable du décodeur dans les ensembles pièges (*Trapping Sets*). Pour y remédier, l'approche adaptative (AGDBF) [17] a introduit un ajustement dynamique des seuils d'inversion. Cependant, cette perturbation globale s'applique à la trame entière sans distinguer l'état individuel de chaque nœud.

Pour affiner la correction, les architectures ont évolué vers le décodeur à diversité (DGDBF) [17]. Cette méthode explore simultanément plusieurs trajectoires de décodage. Si elle permet de contourner des pièges topologiques plus complexes, elle engendre une multiplication de la complexité calculatoire, incompatible avec les contraintes d'empreinte matérielle des systèmes spatiaux embarqués.

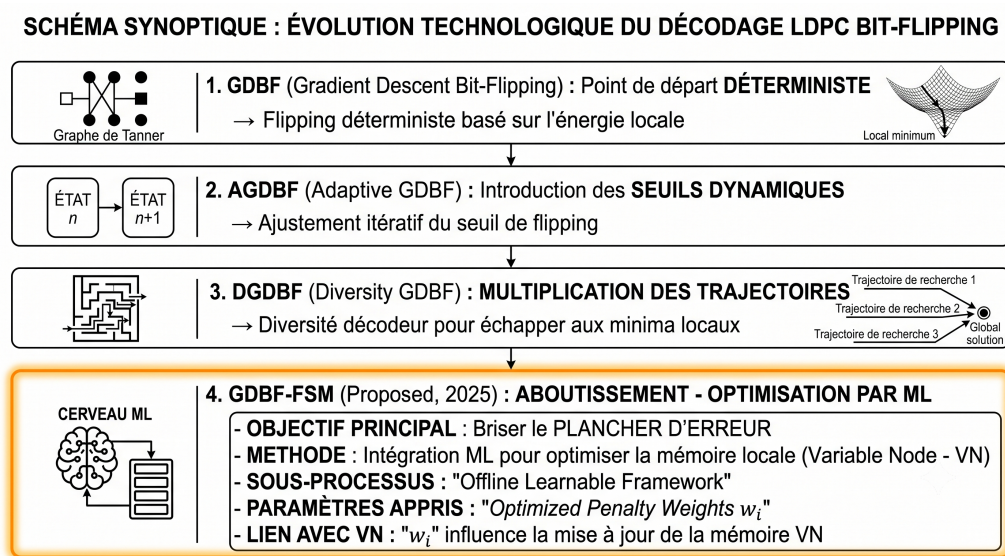


FIGURE 3.1 – Schéma synoptique de l'évolution des algorithmes de décodage, du GDBF classique à l'approche GDBF-FSM.

La Figure 3.1 résume cette progression architecturale, du modèle statique classique vers l'intégration d'une logique à états finis.

L'algorithme gAD-GDBFwM-wSM [12] résout ce compromis en localisant la prise de décision. Plutôt que de multiplier les calculs globaux, chaque nœud de variable (VN) est doté d'une Machine à États Finis (FSM) qui agit comme une mémoire individuelle de son historique d'oscillation. Grâce à un apprentissage hors-ligne des paramètres via un algorithme génétique, cette mémoire locale permet d'adapter intelligemment l'inertie de chaque bit. Le décodeur parvient ainsi à s'extraire des minima locaux avec une précision purement déterministe et une très faible complexité logique.

3.4 Architecture du décodeur GDBF-FSM

Afin d'appréhender la mécanique interne du décodeur GDBF-FSM (*Gradient Descent Bit-Flipping with Finite State Machine*), il est indispensable de définir rigoureusement le cadre algébrique sur lequel s'appuie l'algorithme [12]. Le fonctionnement s'articule autour d'une boucle itérative strictement déterministe, conçue pour minimiser les ressources logiques sur silicium.

La première étape consiste à initialiser l'état matériel du système à partir de l'observation brute du canal de transmission :

$$\hat{x}_i^{(0)} = \text{sign}(y_i) \quad \text{et} \quad r_i^{(0)} = y_i \quad (3.1)$$

Cette équation d'initialisation établit l'état de départ du système à l'instant zéro. La décision bipolaire $\hat{x}_i^{(0)}$ correspond à une quantification immédiate (décision stricte) extraite directement de la polarité du signal reçu. En parallèle, la mémoire locale (le potentiel $r_i^{(0)}$) de la Machine à États Finis est préchargée avec l'amplitude analogique exacte du canal y_i . Physiquement, cette opération dote chaque nœud de variable (VN) d'une inertie de départ proportionnelle à la puissance de l'onde électromagnétique reçue, formant une base de décision qui ne requiert l'activation d'aucun générateur d'aléa.

Une fois l'initialisation achevée, le décodeur vérifie la validité structurelle des estimations en projetant les décisions vers les nœuds de contrôle (CN) :

$$S_j^{(\ell)} = \prod_{k \in N(j)} \hat{x}_k^{(\ell-1)} \quad (3.2)$$

Cette opération évalue les équations de parité en déterminant le syndrome bipolaire de chaque nœud de contrôle j . Le recours au produit logique garantit que si une équation abrite un nombre impair de bits erronés (de valeur -1), le syndrome résultant sera strictement négatif, signifiant une violation de la contrainte physique. Cette projection transforme les relations globales du code LDPC en métriques d'erreurs locales, ciblant

précisément les régions topologiques du graphe nécessitant une intervention.

Les informations renvoyées par les nœuds de contrôle permettent ensuite d'évaluer la fonction d'énergie objective au niveau de chaque nœud de variable i :

$$E_i^{(\ell)} = w_1 \cdot y_i \hat{x}_i^{(\ell-1)} + w_2 \sum_{j \in M(i)} S_j^{(\ell)} + m_{\mu_i} \quad (3.3)$$

Cette équation centrale fusionne toutes les données disponibles pour quantifier l'instabilité locale d'un nœud. Le premier terme évalue la corrélation avec le signal brut (pondéré par w_1), tandis que le second agglomère les syndromes adjacents (pondérés par w_2) pour vérifier le respect des règles de parité. L'innovation architecturale majeure de ce modèle réside dans l'intégration additive du terme m_{μ_i} , qui représente l'état de pénalité (momentum) dicté par la FSM locale. D'un point de vue matériel, cette pénalité agit comme une force de rappel dynamique : elle modifie artificiellement le paysage énergétique du nœud, forçant l'algorithme à extraire le bit d'un minimum local (*Trapping Set*) lorsque la structure détecte un phénomène de stagnation infinie.

Suite à cette évaluation énergétique, le système applique une politique de sélection pour identifier les bits nécessitant une correction :

$$\mathcal{F}^{(\ell)} = \{v_i \mid E_i^{(\ell)} \leq E_{min}^{(\ell)} + \delta\} \quad (3.4)$$

Cette équation détermine l'ensemble d'activation $\mathcal{F}^{(\ell)}$ en charge du basculement multi-bits (*Multi-Bit Flipping*). Plutôt que de cibler uniquement l'unique bit possédant l'énergie absolue la plus faible, l'algorithme intègre un seuil de tolérance δ permettant de sélectionner simultanément un groupe de nœuds fortement suspects. D'un point de vue architectural, cette sélection parallèle accélère la vitesse de convergence globale. Elle autorise la correction d'erreurs concurrentes sur un seul cycle d'horloge, un prérequis fondamental pour supporter les débits élevés imposés par les standards de télécommunications spatiaux.

Enfin, pour les nœuds ciblés par la sélection ou modifiés par la dynamique de la trame, la Machine à États Finis intervient pour actualiser leur mémoire :

$$r_i^{(\ell)} = r_i^{(\ell-1)} + \text{sign}(r_i^{(\ell-1)}) \cdot s_{v_i} \quad \text{suivi de} \quad \hat{x}_i^{(\ell)} = \text{sign}(r_i^{(\ell)}) \quad (3.5)$$

Cette règle logique gouverne la mise à jour de la mémoire interne (le potentiel). Au lieu d'inverser directement le signe d'un bit identifié comme erroné, la FSM accumule l'historique des états en ajoutant ou en soustrayant une valeur d'ajustement s_{v_i} (dictée par l'état courant de l'automate pour le nœud v_i) au potentiel existant. Physiquement, ce mécanisme opère comme un filtre d'inertie : un bit identifié comme suspect verra son potentiel s'effriter graduellement jusqu'à changer de signe, déclenchant alors seulement son

renversement effectif $\hat{x}_i^{(\ell)}$. Cette inertie programmable élimine le phénomène de boucles d'oscillations destructrices inhérent aux algorithmes classiques sans mémoire.

L'ensemble de ces opérations itératives, incluant l'évaluation des potentiels et la logique de mise à jour de la machine à états, est formellement synthétisé par le pseudo-code algorithmique suivant.

Algorithm 1 gAD-GDBFwM with Finite State Machine in one component decoder
Input: $\mathbf{y} = (y_1, y_2, \dots, y_N) \in \mathbb{R}^N$, $\mathbf{r}_{in}^{(t)}$, $\boldsymbol{\mu}_{in}^{(t)}$, $\mathbf{P}_{in}^{(t)}$, δ Output: $\hat{\mathbf{x}} = (\hat{x}_1^{(\ell)}, \hat{x}_2^{(\ell)}, \dots, \hat{x}_N^{(\ell)}) \in \{\pm 1\}^N$, $\mathbf{r}_{out}^{(t)}$, $\boldsymbol{\mu}_{out}^{(t)}$, $\mathbf{P}_{out}^{(t)}$
Initialization: $\ell = 1, \mathbf{r}^{(1)} = \mathbf{r}_{in}^{(t)}, \boldsymbol{\mu} = \boldsymbol{\mu}_{in}^{(t)}, \mathbf{P} = \mathbf{P}_{in}^{(t)}$
$\hat{x}_i^{(1)} = \text{sign}(r_i^{(1)}), i = 1, 2, \dots, N$ $s_j^{(1)} = \prod_{i \in \mathcal{Q}(c_j)} \hat{x}_i^{(1)}, j = 1, 2, \dots, M$ while $(\ell \leq L_{\max, t})$ or $(s_j^{(\ell)} = 1, \forall j)$ do $E_{\min}^{(\ell)} = +\infty$ for $i = 1, \dots, N$ do $E_i^{(\ell)} = w_1^{(\ell)} y_i \hat{x}_i^{(\ell)} + w_2^{(\ell)} \sum_{j \in \mathcal{P}(v_i)} s_j^{(\ell)} + m_{\mu_i}$ $E_{\min}^{(\ell)} = \min\{E_{\min}^{(\ell)}, E_i^{(\ell)}\}$ end for $\mathcal{F}^{(\ell)} = \{v_i E_i \leq E_{\min} + \delta\}$ update $\boldsymbol{\mu}$ accordingly to $\mathcal{F}^{(\ell)}$ update $\mathbf{P}$ accordingly to $\mathcal{F}^{(\ell)}$ and state machine $r_i^{(\ell+1)} = r_i^{(\ell)} + s_{v_i} \text{sign}(r_i^{(\ell)}), i = 1, 2, \dots, N$ apply rule θ if necessary and update $r_i^{(\ell+1)}, i = 1, 2, \dots, N$ $\hat{x}_i^{(\ell+1)} = \text{sign}(r_i^{(\ell+1)}), i = 1, 2, \dots, N$ $s_j^{(\ell+1)} = \prod_{i \in \mathcal{Q}(c_j)} \hat{x}_i^{(\ell+1)}, j = 1, 2, \dots, M$ $\ell = \ell + 1$ end while $\mathbf{r}_{out}^{(t)} = \mathbf{r}^{(\ell)}, \boldsymbol{\mu}_{out}^{(t)} = \boldsymbol{\mu}, \mathbf{P}_{out}^{(t)} = \mathbf{P}$

FIGURE 3.2 – Algorithme de décodage gAD-GDBFwM FSM [12].

3.4.1 Définition de la Machine à États Finis

Pour moduler dynamiquement l'inertie de chaque nœud de variable (VN), l'architecture intègre un automate à états finis (FSM) local, noté $\mathcal{S}$. Chaque état $S \in \mathcal{S}$ délivre une valeur d'ajustement matérielle spécifique s , qui est additionnée au potentiel du nœud. Cet automate, composé de $|\mathcal{S}|$ états, est logiquement structuré en trois groupes [12] :

- Un ensemble d'états positifs, noté $S_+ = \{S_{+,1}, S_{+,2}, \dots, S_{+,\eta}\}$, générant des incréments $s_+ \geq 0$.
- Un ensemble d'états négatifs, défini par $S_- = \{S_{-,1}, S_{-,2}, \dots, S_{-,\zeta}\}$, forçant des décréments $s_- < 0$.
- Un état de repos ou neutre, représenté par S_0 , n'appliquant aucune modification au potentiel ($s_0 = 0$).

Lors de l'initialisation du circuit, la machine à états est placée dans une configuration de départ prédéfinie, désignée par le paramètre S_{start} .

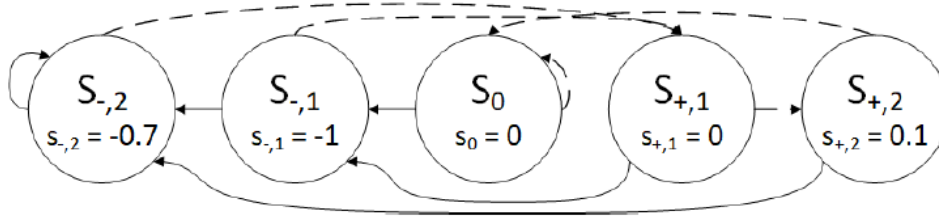


FIGURE 3.3 – Diagramme de transition des états de la FSM locale (Adapté de Milojković et al., 2025 [12]).

La Figure 3.3 modélise la logique séquentielle de cet automate. Les transitions d'états sont strictement conditionnées par l'évaluation énergétique du nœud à chaque cycle d'horloge. Un trait continu indique la transition effectuée si le nœud est mis en évidence (*highlighted*) par le décodeur, c'est-à-dire s'il est ciblé pour une potentielle inversion. Un trait discontinu trace le chemin de transition si le nœud n'est pas sélectionné.

D'un point de vue architectural, l'automate implémente un filtrage temporel de la décision. Un déplacement vers la gauche du diagramme (progression depuis S_0 vers $S_{-,ζ}$) correspond à une phase d'*Hésitation*. Le système accumule des doutes sur la fiabilité du bit et réduit son potentiel pour le rapprocher de son point de basculement. À l'inverse, une progression vers la droite du graphe caractérise une phase de *Détermination*, confirmant la robustesse de l'estimation courante [12].

Concrètement, la logique matérielle de mise à jour s'exécute ainsi : si le système ne cible pas un nœud lors d'un cycle, la polarité de son potentiel doit être préservée. Si cette non-sélection se répète, la fiabilité de la variable est validée par le système. L'automate progresse alors vers la droite jusqu'à atteindre l'état S_0 après η cycles consécutifs. Dans cet état de repos, l'accumulation est suspendue pour économiser l'énergie de commutation des registres, et le potentiel reste figé tant que le nœud n'est pas de nouveau mis en évidence [12]. L'inversion effective de la valeur du bit en sortie de registre ne s'exécute que lorsque la somme algébrique des ajustements s finit par forcer le potentiel global à inverser son bit de signe.

3.4.2 Exemple de résolution d'un ensemble piège (*Trapping Set*)

Afin d'illustrer concrètement la supériorité de l'approche par machine à états finis, il est pertinent d'analyser le comportement matériel du décodeur face à une singularité topologique. Milojković et al. [12] modélisent un motif d'erreurs caractéristique, généré

sur un canal binaire symétrique (BSC), qui met systématiquement en échec l'algorithme GDBF classique en formant un ensemble piège (*Trapping Set*).

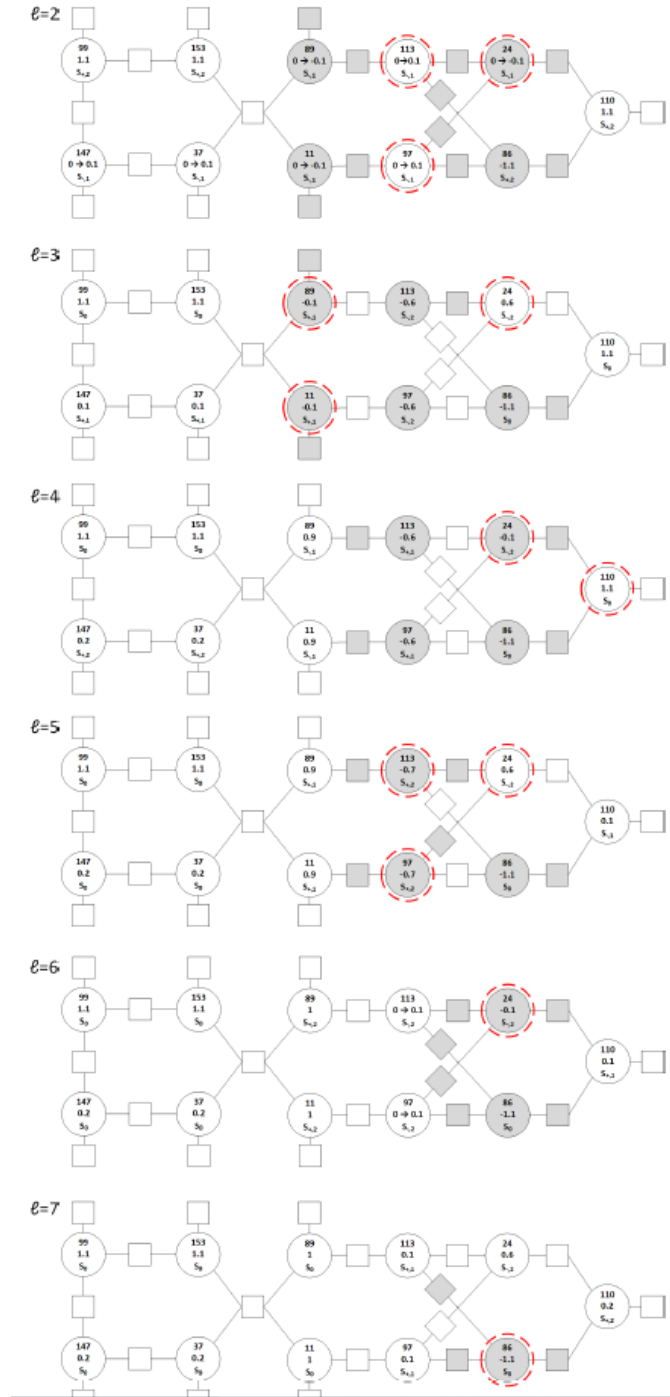


FIGURE 3.4 – Évolution des paramètres d'un ensemble piège sous l'action de la FSM locale (Adapté de Milojković et al., 2025 [12]).

L'évaluation du GDBF conventionnel sur ce motif révèle une instabilité algorithmique sévère. Après seulement trois itérations de traitement, le système s'enferme dans une boucle d'oscillation infinie. Le décodeur alterne perpétuellement l'inversion de deux nœuds

de variable spécifiques (identifiés comme les variables v_{24} et v_{110}), se montrant mathématiquement incapable de stabiliser le syndrome pour faire converger le mot de code [12].

L'intégration de la FSM locale rompt cette boucle d'oscillation par un filtrage temporel des décisions. L'analyse de la quatrième itération ($\ell = 4$) de ce scénario illustre parfaitement cette mécanique. À ce cycle d'horloge précis, le nœud fortement suspect v_{110} est de nouveau ciblé par l'évaluation énergétique. Une architecture sans mémoire l'aurait immédiatement inversé, perpétuant ainsi le blocage. À l'inverse, l'automate FSM décroît son potentiel interne sans pour autant forcer le franchissement du seuil de polarité. Le bit de signe restant inchangé, l'inversion effective de la variable est bloquée [12].

Cette inertie temporelle force le décodeur à explorer de nouvelles trajectoires de correction. Couplée à l'application de la règle θ (qui empêche le registre de se figer sur un potentiel strictement nul en appliquant un léger décalage opposé), cette mécanique modifie artificiellement le paysage énergétique du graphe. Le système parvient ainsi à se stabiliser et à résoudre totalement ce piège topologique en exactement huit itérations [12]. Ce cas de validation prouve la capacité de l'architecture à traiter des configurations d'erreurs complexes par une approche purement déterministe, validant ainsi son efficacité matérielle.

3.4.3 Framework d'Apprentissage et d'Optimisation des Paramètres

Afin de déterminer les valeurs optimales régissant l'automate, la méthodologie retenue s'appuie sur un découplage strict entre la phase d'entraînement et la phase d'inférence matérielle [12]. L'intégralité de la charge calculatoire inhérente à la recherche des paramètres est déportée vers un processus d'apprentissage hors-ligne (*offline learning*), exécuté sur des serveurs de calcul (CPU/GPU). Une fois la convergence atteinte, les paramètres optimaux obtenus sont figés et implémentés sous forme de constantes physiques dans le circuit logique (FPGA ou ASIC). Cette ségrégation garantit que le décodeur opérant en temps réel conserve l'extrême légèreté et la rapidité d'exécution propres aux architectures à décision stricte, sans subir la moindre surcharge liée aux opérations d'apprentissage.

La première étape de cette conception consiste à générer une base de données d'erreurs hautement ciblée (*Error Log*). Plutôt que d'optimiser le système sur un bruit aléatoire trivial, une simulation de type Monte-Carlo est exécutée en utilisant un décodeur GDBF classique soumis à un canal fortement bruité. Cette simulation cible spécifiquement la région du plancher d'erreur (*error floor*). Le système isole et collecte exclusivement les motifs d'erreurs insolubles (*uncorrectable error patterns*) qui provoquent le blocage de l'algorithme déterministe standard. Ces configurations pathologiques, majoritairement constituées d'ensembles pièges (*Trapping Sets*) tenaces, constituent le jeu de données

d'entraînement [12].

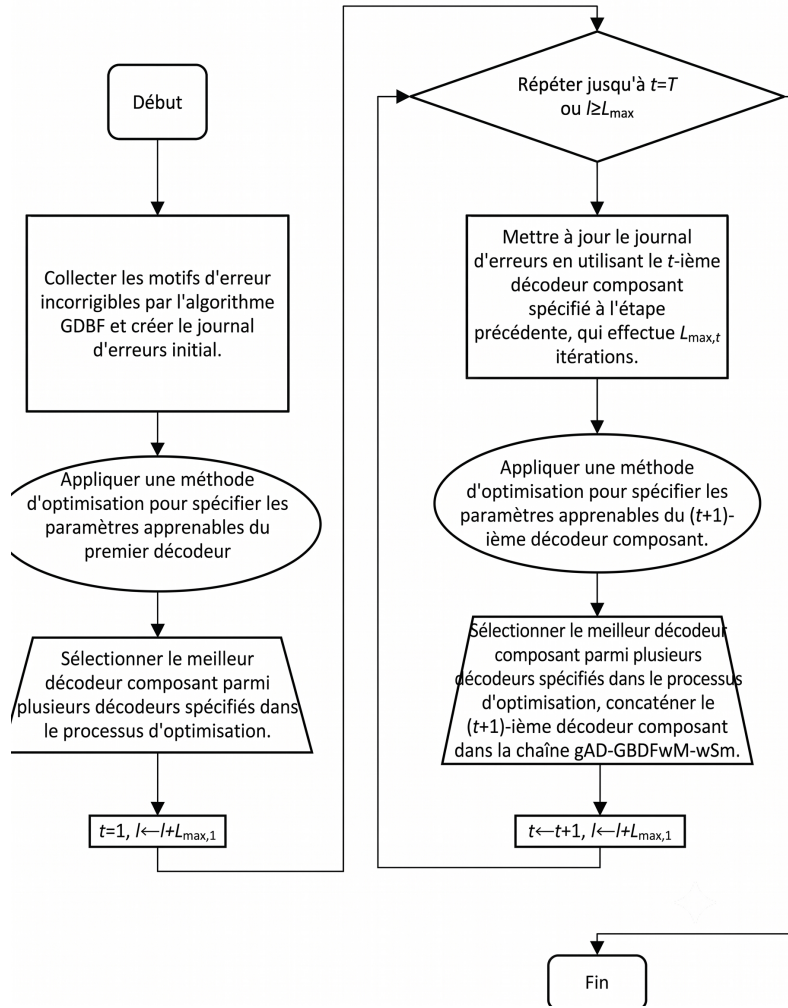


FIGURE 3.5 – Organigramme du processus d'apprentissage hors-ligne pour la collecte des erreurs et l'optimisation des paramètres du décodeur [12].

À partir de ce journal d'erreurs, un Algorithme Génétique (GA) explore l'espace des solutions pour déterminer la configuration optimale du décodeur. Contrairement aux approches heuristiques manuelles, le processus vise à calibrer conjointement et avec une précision discrète les pondérations globales d'énergie (w_1 et w_2), le vecteur de pénalité de momentum (m), ainsi que les valeurs d'ajustement spécifiques à chaque état de la FSM (s). Les paramètres d'architecture (tels que la marge δ ou la règle θ) sont quant à eux fixés empiriquement en amont [12].

La fonction de coût (*loss function*) guidant cet apprentissage est directe : son objectif mathématique est de maximiser le taux d'adaptation (*Fitness Rate*), défini comme le pourcentage de trames corrigées avec succès exclusivement sur cette base de données d'erreurs complexes. Le système génère ainsi une configuration paramétrique sur mesure, capable d'adapter l'inertie du décodeur pour briser les symétries topologiques les plus sévères du graphe de Tanner [12].

L'enchaînement logique de ce processus d'entraînement hors-ligne, allant de la capture des syndromes d'erreurs jusqu'à la sélection de la configuration matérielle finale, est modélisé par l'organigramme de la Figure 3.5.

3.5 Étude des performances de l'algorithme

Afin de caractériser l'efficacité de l'architecture GDBF-FSM, les performances en correction d'erreurs sont évaluées sur deux environnements de transmission standards : le canal binaire symétrique (BSC) et le canal à bruit blanc gaussien additif (AWGN) [12]. L'analyse porte sur deux métriques fondamentales : l'évolution du Taux d'Erreur Trame (FER) face à la dégradation du signal (caractérisée par la probabilité de croisement α ou le rapport signal-sur-bruit), et la vitesse de convergence mesurée par l'impact du nombre maximum d'itérations autorisées (L_{max}).

3.5.1 Matrice de Tanner

La première phase d'évaluation s'applique à la topologie du code régulier de Tanner (155, 64), défini par des degrés de nœuds $\gamma = 3$ et $\rho = 5$. L'étude de la convergence algorithmique (Figure 3.6) confirme que l'intégration de la machine à états franchit les limites de stagnation structurelles des méthodes à décision stricte classiques. L'analyse de l'évolution du FER en fonction de L_{max} démontre que l'architecture FSM abaisse continuellement le taux d'erreur par un filtrage temporel efficace, là où le décodeur GDBF conventionnel s'enferme dans un blocage définitif prématuré [12].

D'un point de vue physique, cette dynamique confirme l'efficacité de la mémoire locale face aux singularités topologiques du code. La gestion déterministe de l'inertie des nœuds de variable désamorce les boucles d'oscillations infinies causées par les ensembles pièges (*Trapping Sets*). Sur le plan des performances, cela se traduit par une chute drastique du FER et une suppression du plancher d'erreur (*error floor*) à faible niveau de bruit [12].

Comme l'illustre la Figure 3.7, cette optimisation de la logique de décision permet à l'algorithme de surpasser largement les performances des variantes probabilistes (telles que le PGDBF ou le TRGDBF), et ce, sans nécessiter l'intégration coûteuse de générateurs d'aléa matériels sur la puce de silicium. Fait remarquable pour cette topologie spécifique, l'approche GDBF-FSM parvient même à concurrencer et à battre l'algorithme à décision pondérée de référence (*Belief Propagation* - BP) dans la région du plancher d'erreur [12].

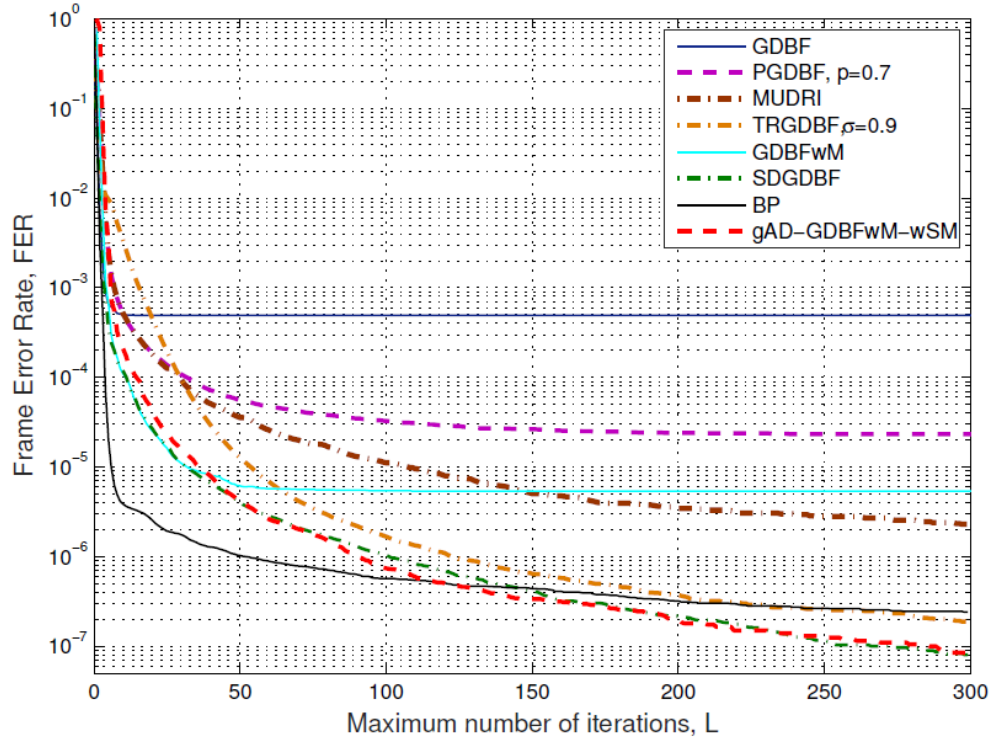


FIGURE 3.6 – Performances en fonction de L_{max} sur canal BSC pour le code de Tanner (155, 64), avec $\gamma = 3$ et $\rho = 5$ [12].

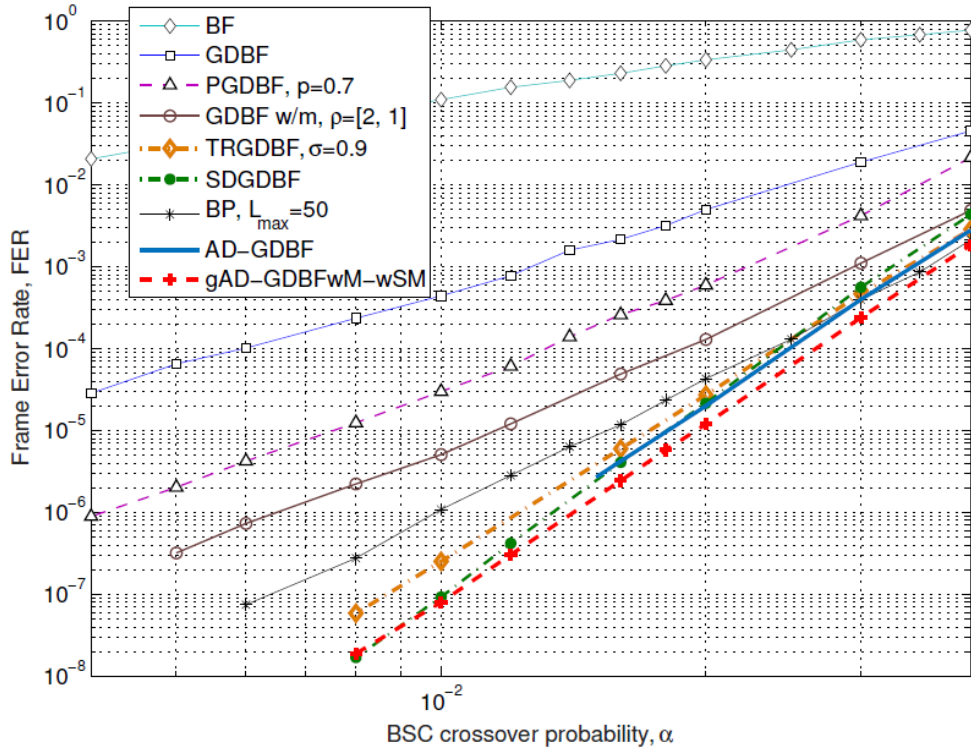


FIGURE 3.7 – Performances comparées des différents décodeurs sur canal BSC pour le code de Tanner (155, 64), avec $\gamma = 3$ et $\rho = 5$ [12].

3.5.2 Matrice i-RISC

L'analyse porte sur une topologie de plus grande dimension via le code régulier i-RISC (1296, 648), défini par des degrés de nœuds $\gamma = 4$ et $\rho = 8$. L'évaluation sur canal BSC illustre la dynamique de convergence en fonction de L_{max} . Dans cette architecture matérielle, chaque incrément de 50 itérations correspond à l'activation d'un décodeur FSM additionnel dans la chaîne de traitement.

L'ajout séquentiel de ces modules entraîne une chute abrupte du FER. Pour $\alpha = 0.03$, l'activation d'un second étage ($L_{max} = 100$) abaisse le taux d'erreur d'une décade complète par rapport à l'étage initial ($L_{max} = 50$) [12]. Cette progression valide le modèle d'apprentissage en cascade : les premiers décodeurs corrigent les erreurs topologiques simples, tandis que les étages subséquents ciblent exclusivement les ensembles pièges résiduels de haute complexité.

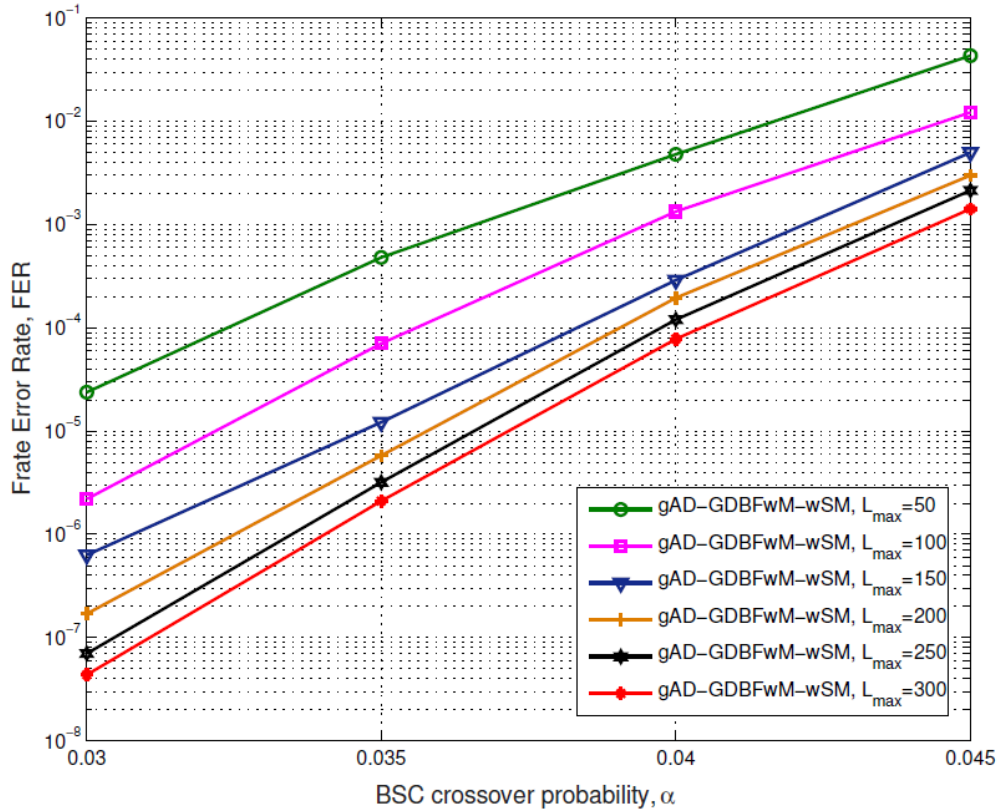


FIGURE 3.8 – Performances en fonction de L_{max} sur canal BSC pour le code i-RISC (1296, 648), avec $\gamma = 4$ et $\rho = 8$ [12].

L'évaluation sur canal AWGN confirme cette dynamique de traitement en cascade et démontre la supériorité de l'architecture face aux modèles probabilistes. La courbe du FER indique qu'une chaîne limitée à quatre décodeurs FSM ($L_{max} = 200$) surpasse les performances de l'algorithme PGDBF avec Momentum paramétré à 300 itérations [12].

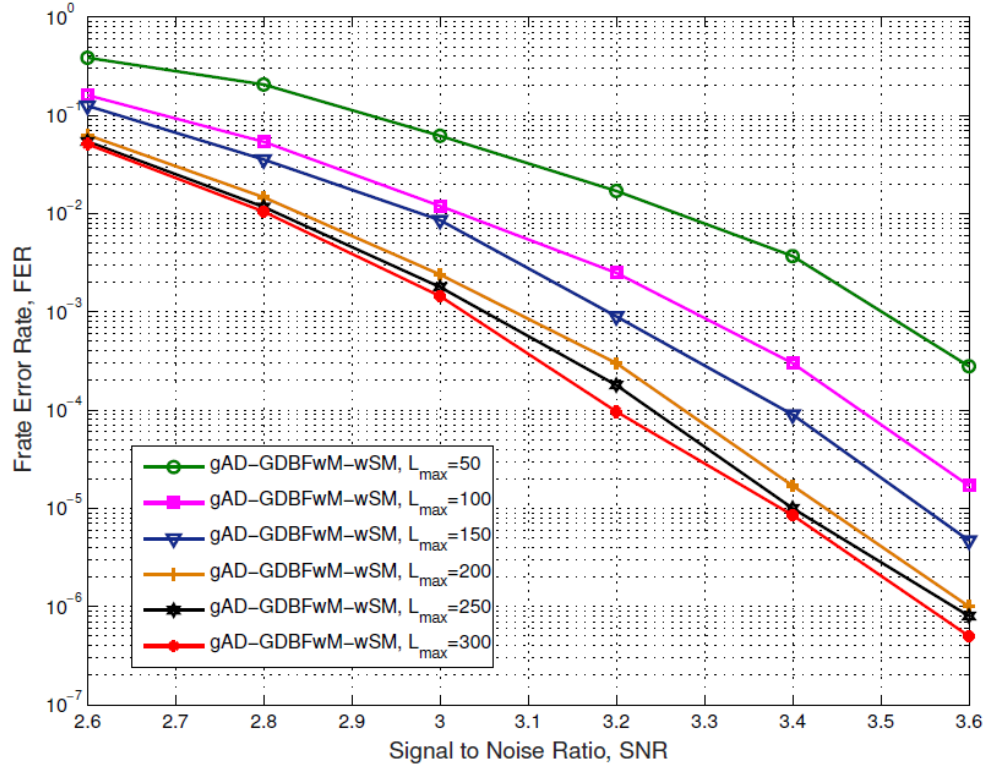


FIGURE 3.9 – Performances en fonction de L_{max} sur canal AWGN pour le code i-RISC (1296, 648), avec $\gamma = 4$ et $\rho = 8$ [12].

Sur le plan de la conception logique, ce résultat valide l'efficacité énergétique du système. L'approche purement déterministe de la FSM, exempte de générateurs d'aléa matériels, garantit un pouvoir de correction supérieur avec un budget d'itérations réduit, minimisant ainsi la latence globale du circuit.

3.6 Conclusion du chapitre

Ce troisième chapitre a formalisé la conception algorithmique et matérielle du décodeur hybride GDBF-FSM. L'étude de cette architecture a mis en évidence les avancées techniques suivantes pour les systèmes de télécommunications embarqués :

- **Filtrage temporel par logique séquentielle** : L'intégration d'une mémoire locale (Machine à États Finis) au niveau de chaque nœud de variable résout la vulnérabilité structurelle des algorithmes à décision stricte classiques face aux minima locaux.
- **Échappement déterministe** : L'optimisation hors-ligne des paramètres de la FSM permet au système d'ajuster dynamiquement l'inertie des bits. Le décodeur parvient ainsi à briser les ensembles pièges sans nécessiter l'implémentation de générateurs d'aléa, très coûteux en surface de silicium.

Fort de ce dimensionnement théorique, le chapitre suivant sera exclusivement

consacré à l'évaluation expérimentale de cette architecture. Nous y détaillerons la modélisation et la simulation logicielle de la chaîne de transmission complète sous l'environnement MATLAB. L'exploitation d'une accélération par processeur graphique (GPU) pour les calculs matriciels nous permettra de simuler massivement les trames et de quantifier rigoureusement, à travers les courbes de taux d'erreur (FER/BER), le gain de performance brut apporté par cette logique adaptative face au GDBF conventionnel.

Chapitre 4

Simulations et évaluation des performances

4.1 Introduction

Ce quatrième chapitre est consacré à l'évaluation expérimentale et à la caractérisation des performances en correction d'erreurs des architectures de décodage étudiées précédemment. L'analyse comparative porte sur trois algorithmes distincts : le *Bit-Flipping* (BF) classique, l'approche par descente de gradient (GDBF) de Wadayama [11], et l'architecture adaptative par machine à états finis (gAD-GDBFwM-wSM) de Milojković et al. [12].

Afin de mener une validation exhaustive de ces modèles, le plan de test s'appuie sur plusieurs topologies de codes. Les performances de référence (*baseline*) des algorithmes déterministes standards (BF et GDBF) sont établies sur la matrice du standard DVB-S2. L'architecture hybride FSM est ensuite éprouvée face à des topologies reconnues pour la complexité de leurs ensembles pièges : la matrice de Tanner (155, 64) et le code i-RISC (1296, 648). L'intégralité des transmissions est modélisée sur des canaux BSC et AWGN.

L'évaluation numérique s'appuie sur la méthode de Monte-Carlo, dimensionnée pour traiter jusqu'à 50 000 trames par point de mesure afin d'assurer une convergence statistique rigoureuse des résultats. Pour supporter la charge calculatoire colossale induite par ces matrices de grande dimension, la chaîne de simulation développée sous MATLAB exploite massivement l'accélération matérielle par processeur graphique (GPU). Cette parallélisation permet de quantifier avec une haute précision les métriques FER et BER, validant ainsi concrètement l'apport de la logique adaptative face aux approches classiques.

4.2 Présentation de la chaîne de simulation

Le modèle de simulation développé sous l'environnement MATLAB est illustré par la Figure 4.1. La génération des trames s'appuie sur une source pseudo-aléatoire produisant des vecteurs d'information de dimension K . Ces blocs subissent un encodage LDPC pour former des mots de code de longueur N , qui sont ensuite mappés via une modulation BPSK bipolaire avant leur transmission sur le canal (AWGN ou BSC).

À la réception, le signal altéré par le bruit est injecté dans l'étage de décodage. Selon le scénario d'évaluation, ce bloc matériel exécute soit un algorithme de référence (le BF classique ou le GDBF de Wadayama [11]), soit l'architecture avancée à machine à états (GDBF-FSM [12]). Le système itère sur le syndrome pour estimer la séquence binaire originelle.

L'évaluation de la robustesse de l'algorithme s'effectue en comparant stricte-

ment le vecteur décodé au message initialement émis. Cette confrontation logique permet de quantifier le Taux d'Erreur Binaire (BER) du système :

$$BER = \frac{\text{Nombre de bits erronés}}{\text{Nombre total de bits transmis}} \quad (4.1)$$

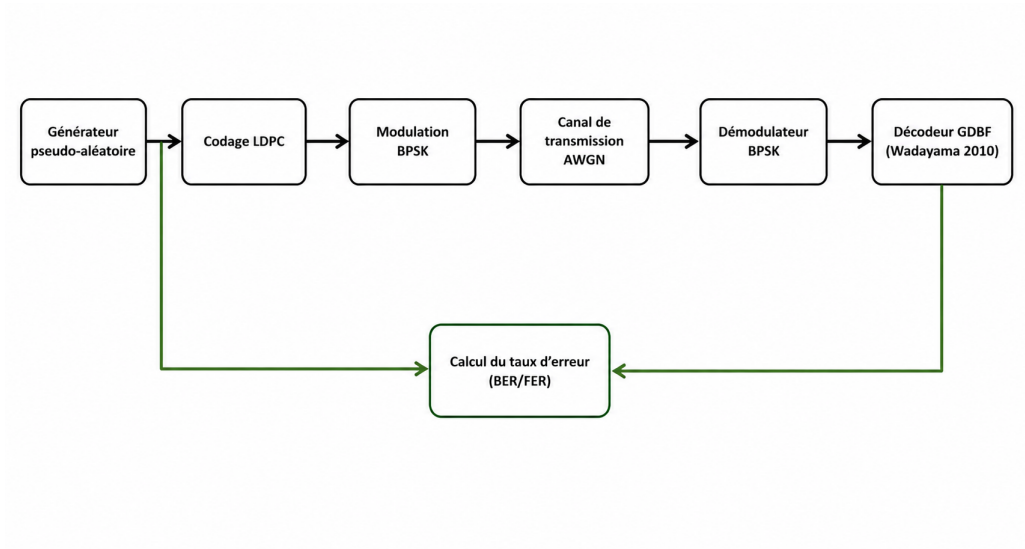


FIGURE 4.1 – Modèle de la chaîne de simulation de transmission LDPC et évaluation des performances.

4.3 Résultats du décodage classique sur la matrice DVB-S2

La première phase de simulation établit les performances de référence (*baseline*) sur la matrice standardisée DVB-S2. Cette topologie est caractérisée par une longueur de bloc $N = 64800$, $K = 32400$ bits d'information, et un rendement $R = 0.5$. L'analyse porte sur l'évolution du BER en fonction du SNR, balayé sur une plage de 2 à 6 dB.

L'analyse de la Figure 4.2 révèle que les courbes de l'algorithme BF (50 itérations) et du GDBF [11] (100 itérations) se superposent quasi intégralement. Sur l'ensemble de la plage de SNR, ces deux décodeurs stagnent à un taux d'erreur compris entre 10^{-1} et 10^{-2} . Les algorithmes ne parviennent pas à franchir la courbe de performance théorique d'une transmission BPSK non codée.

Cette stagnation s'explique par les propriétés topologiques du standard DVB-S2. La matrice de parité, fortement irrégulière, favorise la formation d'ensembles pièges (*Trapping Sets*). Dépourvus de mémoire d'état, les algorithmes à décision

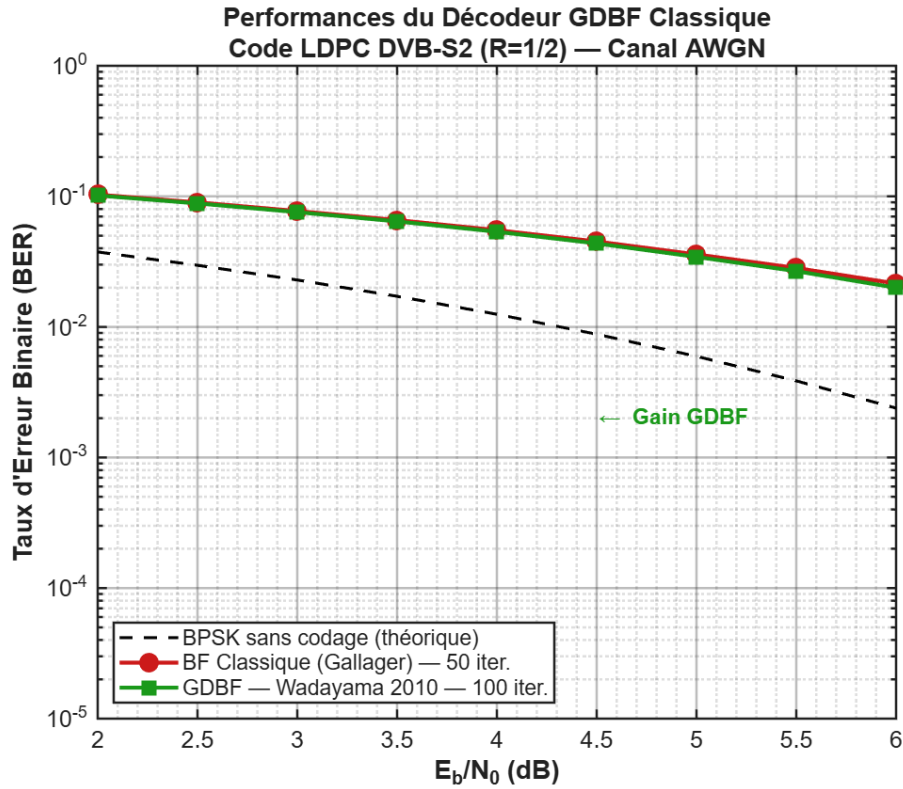


FIGURE 4.2 – Performances des décodeurs BF et GDBF sur la matrice DVB-S2 ($R = 0.5$) sur canal AWGN.

strictes classiques manquent de diversité adaptative pour s'extraire des minima locaux de la fonction d'énergie. En réaction immédiate aux équations insatisfaites, le décodeur inverse simultanément un grand nombre de bits fortement connectés. Le système s'enferme alors dans une boucle d'oscillation infinie, échouant systématiquement à annuler le syndrome.

Ce blocage algorithmique démontre expérimentalement l'incapacité des approches déterministes sans mémoire à traiter des topologies fortement irrégulières par simple descente de gradient. Ce constat valide la nécessité d'intégrer un filtrage temporel via une logique séquentielle (FSM) pour bloquer les inversions prématurées et briser ces boucles infinies. L'évaluation de cette architecture avancée fait l'objet de la section suivante sur les matrices de Tanner et i-RISC.

4.4 Évaluation de l'algorithme gAD-GDBFwM-wSM sur la matrice de Tanner

L'évaluation de l'architecture gAD-GDBFwM-wSM [12] sur la matrice régulière de Tanner (155, 64) est réalisée sur un canal BSC avec une probabilité de croisement fixée à $\alpha = 0.01$. L'objectif de cette simulation est de caractériser la dynamique

de convergence matérielle des décodeurs en mesurant l'évolution du Taux d'Erreur Trame (FER) sur une plage d'exécution s'étendant de 10 à 300 itérations maximales (L_{max}).

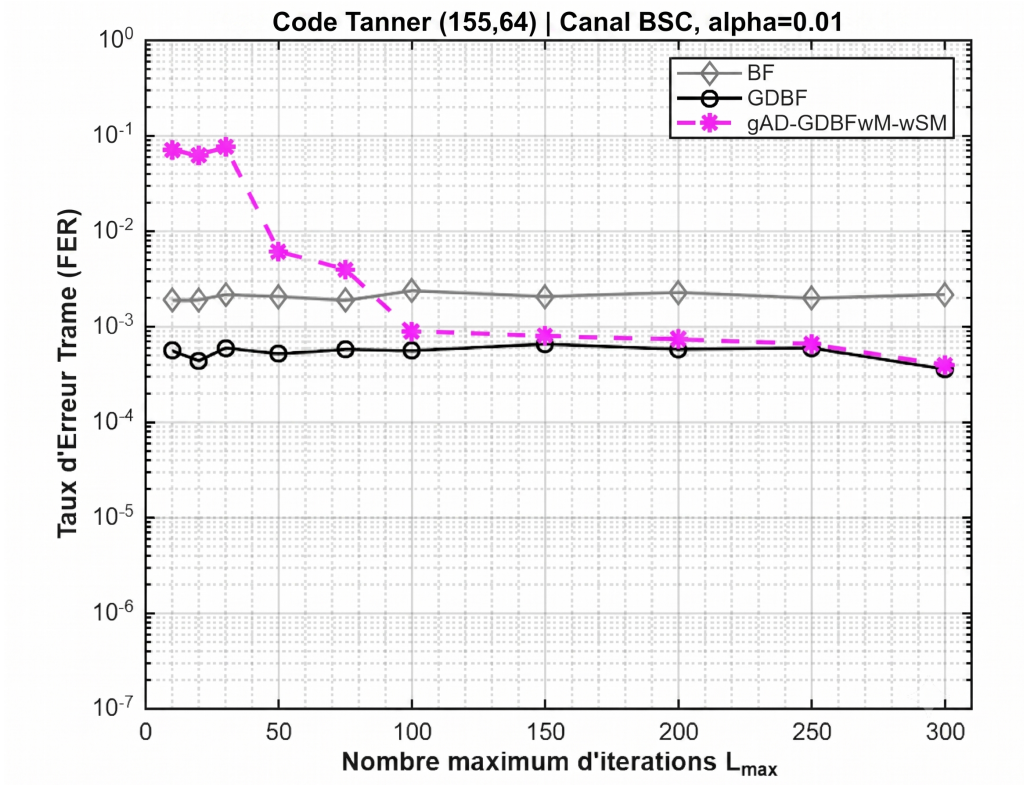


FIGURE 4.3 – Évolution du Taux d'Erreur Trame (FER) en fonction de L_{max} pour le code de Tanner sur canal BSC ($\alpha = 0.01$)

L'analyse de la Figure 4.3 met en évidence la stagnation structurelle du décodeur GDBF classique. La courbe démontre une saturation quasi immédiate de la capacité de correction : le FER plafonne autour de 5.6×10^{-4} dès 10 itérations, et ne s'abaisse qu'à 3.6×10^{-4} à 300 itérations. Ce comportement asymptotique plat prouve expérimentalement que l'algorithme classique sans mémoire s'enferme définitivement dans des minima locaux (*Trapping Sets*). L'augmentation du budget d'itérations n'apporte aucun gain de décodage, confirmant l'inefficacité de l'approche statique de descente de gradient face à ces singularités topologiques [11].

À l'inverse, la trajectoire de convergence du modèle FSM valide le principe d'inertie adaptative. L'observation des 30 premières itérations révèle un FER volontairement maintenu à un niveau élevé (autour de 7.7×10^{-2}). D'un point de vue de l'architecture logique, cette phase correspond au filtrage temporel imposé par la machine à états : le système accumule les incréments de doute dans ses registres internes et bloque physiquement les inversions de bits prématurées afin de ne pas enclencher de boucles d'oscillations destructrices.

Passé ce délai d'accumulation critique (dès l'itération 50), les données montrent une chute abrupte du taux d'erreur. Les potentiels des nœuds franchissent leur seuil de basculement avec certitude, ce qui permet à la courbe de plonger continuellement pour atteindre 4.0010^{-4} à 300 itérations. Cette dynamique matérielle valide formellement l'hypothèse de conception : la cascade d'automates paramétrés parvient à forcer l'échappement hors des minima locaux et à briser le plancher d'erreur des algorithmes heuristiques, à la stricte condition que le séquenceur du circuit alloue un budget d'itérations suffisant pour franchir l'inertie de la mémoire d'états [12].

4.5 Évaluation de l'algorithme gAD-GDBFwM-wSM sur la matrice de i-RISC

L'évaluation porte ensuite sur une topologie de grande dimension, à savoir le code régulier i-RISC ($N = 1296$, $K = 648$). L'augmentation de la taille de la matrice accroît structurellement le nombre d'ensembles pièges (*Trapping Sets*) complexes. Pour soutenir la charge de calcul imposée par la simulation de dizaines de milliers de trames, une accélération matérielle par processeur graphique (GPU) a été exploitée. Cette parallélisation garantit la fiabilité statistique des mesures à travers les deux canaux de transmission (BSC et AWGN).

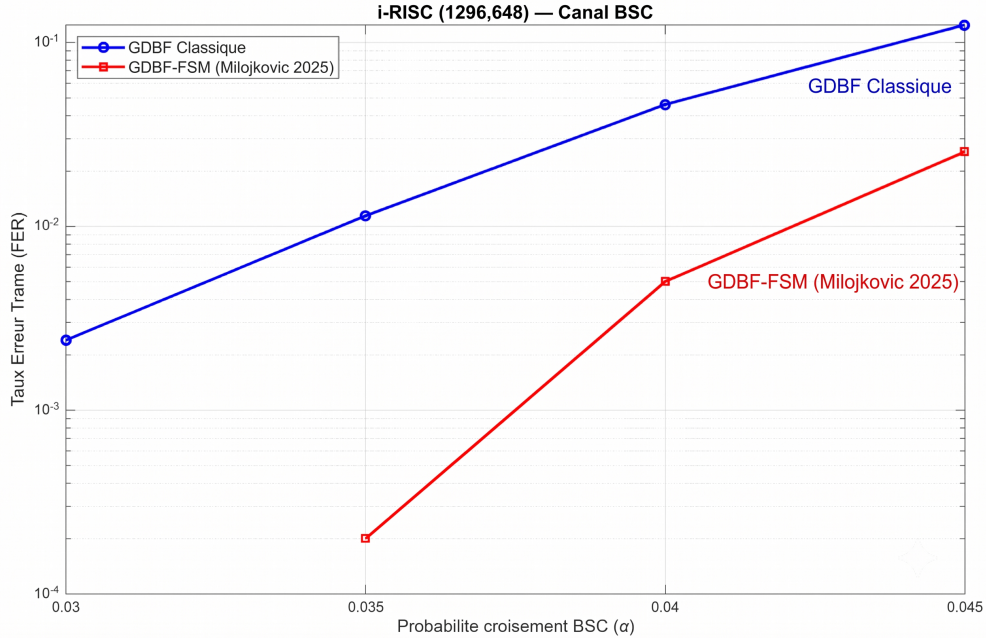


FIGURE 4.4 – Évolution du FER en fonction de la probabilité de croisement α pour le code i-RISC (1296, 648) sur canal BSC.

L'analyse sur le canal BSC, balayé sur une probabilité de croisement α de 0.030 à 0.045 (Figure 4.4), confirme la saturation prématurée des algorithmes classiques sans mémoire. Pour $\alpha = 0.035$, le décodeur GDBF de référence stagne à un FER de 1.1410^{-2} . À l'inverse, le modèle FSM démontre l'efficacité de sa logique adaptative : en forçant l'extraction hors des minima locaux, il abaisse le taux d'erreur à 210^{-4} pour les mêmes conditions de canal.

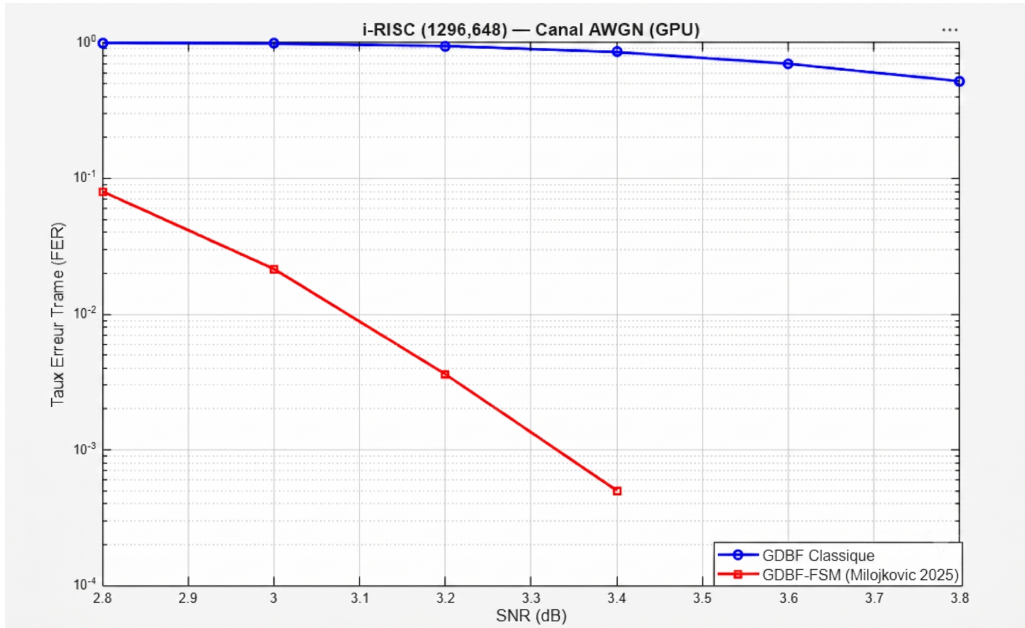


FIGURE 4.5 – Évolution du FER en fonction du SNR pour le code i-RISC (1296, 648) sur canal AWGN.

L'évaluation sur le canal AWGN (SNR de 2.8 à 3.8 dB) conforte cette dynamique matérielle (Figure 4.5). Face aux perturbations analogiques continues, le GDBF classique sature rapidement et présente un plancher d'erreur absolu proche de 10^0 , confirmant son incapacité à résoudre les pièges topologiques complexes. En revanche, l'algorithme GDBF-FSM franchit ces blocages pour générer une chute abrupte du taux d'erreur (*waterfall*), atteignant 510^{-4} à 3.4 dB. Sur ces mêmes scénarios, le décodeur BP, exécuté sur le GPU pour modéliser la borne théorique de correction, affiche un taux d'erreur quasi nul.

Ces résultats valident expérimentalement la viabilité de l'architecture étudiée : l'algorithme FSM parvient à s'approcher du pouvoir de correction optimal du décodage à décision pondérée, tout en maintenant la faible empreinte logique caractéristique des décodeurs à décision stricte.

4.6 Synthèse comparative des résultats

Afin de consolider l'ensemble des résultats expérimentaux présentés dans ce chapitre, le Tableau 4.1 récapitule les performances obtenues sur les trois matrices étudiées (DVB-S2, Tanner, i-RISC), pour les conditions de canal les plus représentatives de chaque scénario.

TABLE 4.1 – Synthèse comparative des performances (FER) des décodeurs étudiés sur les trois matrices LDPC testées.

Matrice	Canal	Condition	GDBF	GDBF-FSM	Gain
DVB-S2 (64800,32400)	AWGN	$E_b/N_0 = 6$ dB	$2,010^{-2}$	–	–
Tanner (155,64)	BSC	$\alpha = 0,01$	$5,610^{-4}$	$4,010^{-4}$	1,4
i-RISC (1296,648)	BSC	$\alpha = 0,035$	$1,1410^{-2}$	$2,010^{-4}$	57
i-RISC (1296,648)	AWGN	$SNR = 3,4$ dB	≈ 1	$5,010^{-4}$	> 2000

L'analyse de ce tableau permet de dégager trois constats majeurs.

Premièrement, sur la matrice DVB-S2, fortement irrégulière, ni le décodeur BF ni le décodeur GDBF classique ne parviennent à franchir la courbe de performance théorique d'une transmission BPSK non codée (Figure 4.2). Cette configuration sert exclusivement de référence *baseline* pour démontrer l'incapacité structurelle des algorithmes sans mémoire d'état à traiter les ensembles pièges des matrices irrégulières.

Deuxièmement, sur le code de Tanner (155,64), le gain apporté par l'architecture GDBF-FSM reste modéré (facteur 1,4) mais significatif sur le plan qualitatif : alors que le GDBF classique plafonne dès la dixième itération sans plus jamais évoluer, le décodeur FSM continue de progresser jusqu'à la trois centième itération, prouvant qu'il échappe effectivement aux minima locaux.

Troisièmement, sur le code i-RISC (1296,648), de dimension plus importante, le gain devient spectaculaire : un facteur 57 sur canal BSC et un facteur supérieur à 2000 sur canal AWGN. Ce résultat confirme que l'apport de la mémoire d'état FSM est d'autant plus déterminant que la taille du code – et donc la complexité topologique des ensembles pièges – augmente.

En définitive, cette synthèse valide la problématique centrale de ce mémoire : l'intégration d'une machine à états finis, dont les paramètres sont optimisés hors-ligne par apprentissage automatique, permet de briser le plancher d'erreur structurel du décodage à décision stricte, sans aucune augmentation de la complexité matérielle du circuit de décodage.

4.7 Conclusion du chapitre

Ce chapitre a été consacré à l'évaluation expérimentale des architectures de décodage sous l'environnement MATLAB. L'analyse comparative de nos résultats de simulation a permis d'établir les constats techniques suivants :

- **Limites de l'approche classique :** L'algorithme GDBF à décision stricte sature rapidement et subit un plancher d'erreur sévère, démontrant son incapacité à résoudre les ensembles pièges (*Trapping Sets*).
- **Efficacité de la logique adaptative :** L'intégration d'une mémoire locale (FSM) force l'extraction du décodeur hors des minima locaux, générant une chute drastique du taux d'erreur (*waterfall*).
- **Nécessité de l'accélération matérielle :** Le déploiement des calculs sur processeur graphique (GPU) s'est avéré indispensable pour traiter massivement les trames et garantir une fiabilité statistique irréprochable.
- **Compromis d'ingénierie optimal :** L'architecture FSM parvient à concurrencer le pouvoir de correction des décodeurs à décision pondérée (*Soft*), tout en conservant la très faible empreinte matérielle d'une décision stricte (*Hard*).

En définitive, ces expérimentations valident formellement la problématique de ce mémoire. Nous avons prouvé que l'optimisation des paramètres d'inversion par une logique d'états finis permet de concevoir un décodeur hautement performant, parfaitement aligné avec les contraintes matérielles des futurs systèmes de télécommunications.

Conclusion Générale

Ce projet de fin d'études a été consacré à l'optimisation algorithmique et matérielle du décodage des codes LDPC. Notre démarche s'est focalisée sur l'amélioration de l'algorithme à décision stricte *Bit-Flipping* en y intégrant des techniques d'apprentissage automatique (Machine Learning).

Dans un premier temps, l'étude théorique de la chaîne de communication et des algorithmes classiques a mis en évidence une limitation structurelle majeure. Les approches déterministes, telles que le GDBF conventionnel,aturent prématurément en se retrouvant bloquées dans des minima locaux induits par les ensembles pièges (*Trapping Sets*). Ce constat a formellement justifié la nécessité d'introduire une logique adaptative pour forcer l'extraction du décodeur, évitant ainsi le recours aux méthodes à décision pondérée (*Soft-Decision*), dont la complexité d'intégration matérielle est prohibitive.

Pour lever ce verrou technologique, nous avons modélisé et évalué une architecture hybride novatrice : l'algorithme gAD-GDBFwM-wSM. Ce modèle dote chaque nœud de variable d'une Machine à États Finis (FSM) dont les paramètres de transition et d'inertie ont été optimisés hors-ligne par un Algorithme Génétique. Cette approche permet de guider intelligemment les inversions de bits en fonction de l'historique d'oscillation du graphe.

La validation de cette architecture a été entièrement menée par voie de simulation. Ces campagnes de simulations intensives, réalisées sous MATLAB avec une accélération par processeur graphique (GPU), ont ciblé le code de Tanner (155, 64) ainsi que le code de grande dimension i-RISC (1296, 648) sur des canaux BSC et AWGN.

Les résultats obtenus démontrent que l'algorithme adaptatif constitue un compromis d'ingénierie optimal. Le décodeur FSM parvient à briser le plancher d'erreur inhérent aux méthodes classiques et génère une chute drastique du taux d'erreur (*waterfall*), s'approchant ainsi des performances des algorithmes complexes à déci-

sion souple, tout en garantissant une très faible complexité matérielle et une grande rapidité de traitement.

En définitive, ce travail prouve que l’optimisation hors-ligne des paramètres d’inversion par apprentissage automatique transcende les limites historiques du *Bit-Flipping*. Bien que ce projet apporte une réponse concrète aux contraintes de décodage à très haut débit, il ouvre également la voie à plusieurs perspectives de recherche :

- **Prototypage matériel** : Considérer l’implémentation de l’architecture FSM (gAD-GDBFwM-wSM) sur une cible FPGA ou ASIC afin de valider expérimentalement son efficacité énergétique, son empreinte silicium et son débit effectif en temps réel.
- **Codes non-binaires** : Étudier la faisabilité mathématique et matérielle de l’extension de cette logique d’états finis au décodage des codes LDPC non-binaires, réputés pour leur très haut pouvoir de correction.
- **Intégration systémique** : Adapter et dimensionner cette architecture pour répondre aux exigences strictes des futurs standards de télécommunications, tels que la 6G ou les liaisons optiques satellitaires à ultra-haut débit.

Bibliographie

- [1] H. Nyquist, “Certain factors affecting telegraph speed,” *Bell System Technical Journal*, vol. 3, pp. 324–346, 1924.
- [2] R. Hartley, “Transmission of information,” *Bell System Technical Journal*, vol. 7, 1928.
- [3] C. E. Shannon, “A mathematical theory of communication,” *Bell System Technical Journal*, vol. 27, pp. 379–423, 1948.
- [4] V. Pignoly, *Étude de codes LDPC pour applications spatiales optiques et conception des décodeurs associés*. PhD thesis, Université de Bordeaux, 2021.
- [5] K. L. Du and M. N. S. Swamy, *Wireless Communication Systems*. New York : Cambridge University Press, 2009.
- [6] I. Achouri, “Implémentation sur fpga d’un décodeur ldpc pour les communications sans fils,” mémoire d’ingénieur d’état, École Nationale Polytechnique, Alger, 2018.
- [7] A.-M. Cuc, F. L. Morgoş, and C. Grava, “Performance analysis of turbo codes, ldpc codes, and polar codes over an awgn channel in the presence of inter symbol interference,” *Sensors*, vol. 23, no. 4, p. 1942, 2023.
- [8] R. G. Gallager, “Low-density parity-check codes,” *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, 1962.
- [9] D. J. C. MacKay and R. M. Neal, “Near shannon limit performance of low density parity check codes,” *Electronics Letters*, vol. 33, no. 6, pp. 457–458, 1997.
- [10] R. Tanner, “A recursive approach to low complexity codes,” *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 533–547, 1981.
- [11] T. Wadayama, K. Nakamura, M. Yagita, Y. Funahashi, S. Usami, and I. Takumi, “Gradient descent bit flipping algorithms for decoding ldpc codes,” *IEEE Transactions on Communications*, vol. 58, no. 6, pp. 1610–1614, 2010.

- [12] J. Milojković, S. Brkić, P. Ivaniš, and B. Vasić, “Generalized adaptive diversity gradient descent bit-flipping with a finite state machine,” *Entropy*, vol. 27, no. 1, p. 49, 2025.
- [13] N. Savijoki, “Optimizing ldpc decoding with machine learning using learnable weights,” Master’s thesis, Tampere University, 2025.
- [14] O. A. Rasheed, P. Ivaniš, and B. Vasić, “Fault-tolerant probabilistic gradient-descent bit flipping decoder,” *IEEE Communications Letters*, vol. 18, no. 9, pp. 1487–1490, 2014.
- [15] X. Wu, M. Jiang, and C. Zhao, “Decoding optimization for 5g ldpc codes by machine learning,” *IEEE Access*, vol. 6, pp. 50179–50186, 2018.
- [16] M. Carpi, C. Schlegel, and A. D. Santos, “Reinforcement learning for channel coding : Learned bit-flipping decoding,” in *57th Annual Allerton Conference on Communication, Control, and Computing*, 2019.
- [17] S. Brkić, P. Ivaniš, and B. Vasić, “Adaptive gradient descent bit-flipping diversity decoding,” *IEEE Communications Letters*, vol. 26, no. 10, pp. 2257–2261, 2022.
- [18] A. Shokrollahi, “Ldpc codes : An introduction,” in *Coding, Cryptography and Combinatorics*, Basel : Birkhäuser, 2004.
- [19] R. G. Gallager, “Low-density parity-check codes,” *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, 1962.
- [20] D. J. C. MacKay, “Good error-correcting codes based on very sparse matrices,” *IEEE Transactions on Information Theory*, vol. 45, no. 2, pp. 399–431, 1999.